Matrix product operators, matrix product states, and ab initio density matrix renormalization group algorithms

Garnet Kin-Lic Chan, Anna Keselman, Naoki Nakatani, Zhendong Li, and Steven R. White

Citation: J. Chem. Phys. **145**, 014102 (2016); doi: 10.1063/1.4955108
View online: http://dx.doi.org/10.1063/1.4955108
View Table of Contents: http://aip.scitation.org/toc/jcp/145/1
Published by the American Institute of Physics

Articles you may be interested in

The ab-initio density matrix renormalization group in practice
J. Chem. Phys. **142**, 034102034102 (2015); 10.1063/1.4905329

RIGHTSLINK

Matrix product operators, matrix product states, and *ab initio* density matrix renormalization group algorithms

Garnet Kin-Lic Chan,[1] Anna Keselman,[2,3] Naoki Nakatani,[4] Zhendong Li,[1] and Steven R. White[3]
[1]*Department of Chemistry, Princeton University, Princeton, New Jersey 08544, USA*
[2]*Department of Condensed Matter Physics, Weizmann Institute of Science, Rehovot 76100, Israel*
[3]*Department of Physics and Astronomy, University of California, Irvine, California 92697-4575, USA*
[4]*Institute for Catalysis, Hokkaido University, Kita 21 Nishi 10, Sapporo, Hokkaido 001-0021, Japan*

(Received 16 May 2016; accepted 20 June 2016; published online 5 July 2016)

Current descriptions of the *ab initio* density matrix renormalization group (DMRG) algorithm use two superficially different languages: an older language of the renormalization group and renormalized operators, and a more recent language of matrix product states and matrix product operators. The same algorithm can appear dramatically different when written in the two different vocabularies. In this work, we carefully describe the translation between the two languages in several contexts. First, we describe how to efficiently implement the *ab initio* DMRG sweep using a matrix product operator based code, and the equivalence to the original renormalized operator implementation. Next we describe how to implement the general matrix product operator/matrix product state algebra within a pure renormalized operator-based DMRG code. Finally, we discuss two improvements of the *ab initio* DMRG sweep algorithm motivated by matrix product operator language: Hamiltonian compression, and a sum over operators representation that allows for perfect computational parallelism. The connections and correspondences described here serve to link the future developments with the past and are important in the efficient implementation of continuing advances in *ab initio* DMRG and related algorithms. *Published by AIP Publishing.* [http://dx.doi.org/10.1063/1.4955108]

I. INTRODUCTION

The density matrix renormalization group (DMRG), introduced by White,[1,2] is now more than two decades old. Although originally presented as a computational framework for one-dimensional lattice systems, in the last decade many of its most interesting applications have been to a much broader class of problems. In the context of quantum chemistry, it was recognized early on that, as a non-perturbative method, the DMRG could be a useful tool to replace configuration interaction. With the advent of efficient *ab initio* algorithms in the early 2000's,[3–9] the DMRG has since established itself as an indispensable part of the toolkit of quantum chemistry, especially in problems requiring the accurate treatment of strongly correlated electrons.[4–41]

The conceptual framework of the DMRG has further greatly expanded and deepened in the last decade. In the early 2000's, it became clear that the power of DMRG-like algorithms originates from the "matrix product" structure of the ansatz[42,43] which expresses the low entanglement nature of one-dimensional low-energy quantum eigenstates, such as the ground-state. This entanglement perspective made it possible to expand the ideas of the DMRG into new domains: matrix product operator representations,[44–47] time-evolution,[48–50] infinite systems,[51–53] finite temperatures,[44,54] and higher-dimensions,[46,55–60] to name a few. Beyond computation, the language of matrix product and tensor network states is now widely used to reason about the structure of many-particle quantum states.[45,46,57,61] Within this greatly expanded setting, DMRG is often taken to be synonymous with the sweep-like algorithms commonly used with matrix product states (MPSs) and matrix product operators (MPOs), e.g., "finite-temperature DMRG," "time-dependent DMRG," and "infinite DMRG," while the term "tensor network" embodies the wider class of representations and algorithms associated with higher dimensions.

Early *ab initio* DMRG work focused on how to efficiently implement energy optimization and compute expectation values,[4–8,10,14,16] such as reduced density matrices.[20,22,26,41] These expectation value computations are performed via a sweep algorithm that proceeds through orbitals one-by-one. In the *ab initio* context, the key step is to identify and construct efficiently the appropriate renormalized operators as one proceeds through the sweep. This concept of renormalized operators arises naturally within the renormalization group framework within which the DMRG was originally proposed. In modern day MPO/MPS parlance, however, renormalized operators are simply the computational intermediates corresponding to partial traces of the operator (MPO) with the bra and ket states (MPS) as one includes successive orbitals in a sweep.[61] In an expectation value computation (or optimization), only these partial traces of the MPO are required, and the explicit MPO itself never needs to appear. Thus the original implementations of the *ab initio* DMRG, which focus on renormalized operator-based computation, are not structured around explicit MPO's but rather the renormalized operators and matrix product

RIGHTSLINK

states. We refer to these original implementations as "pure renormalized operator-based" DMRG implementations.

Within the MPO/MPS setting, it is of course natural to implement codes where MPO's appear explicitly. It is important to emphasize that using MPO's in a code does not in itself change the *ab initio* DMRG algorithm. The most efficient serial formulation of expectation value computation (without further approximation) remains to use the MPO and the bra and ket MPS to build the renormalized operators, in precisely the same manner as in the original *ab initio* DMRG. However, having explicit MPO's in the code is useful in connecting to the modern notation and graphical language of MPO's and MPS. We will refer to DMRG programs organized around explicit MPO representations as "MPO-based" DMRG implementations. These MPO-based *ab initio* DMRG implementations have been carried out by several groups, including Murg *et al.*,[58] the authors,[62] and Keller *et al.*[63–65] The implementations typically rely on general MPO/MPS libraries, such as ITensor,[66] MPSXX,[62] and Alps.[67] The implementations have been used in publications and some are freely available. However, with the exception of that of Keller *et al.*[64,65] they have not previously been described in detail in the literature.

The computational steps of an MPO-based implementation are essentially the same as in the traditional "renormalized operator-based" DMRG implementation. However, while the mapping between renormalized operators, and explicit MPO/MPS representations, is well-known in general terms to DMRG practitioners, the lack of an explicit translation between quantities appears as a source of confusion in the wider quantum chemistry community. This is because the language involved in MPO-based implementations and the pure renormalized operator-based implementations can appear very different. For example, in the description of the DMRG algorithm by Keller *et al.* in Refs. 64 and 65, the connection to the identical quantities and operations in the original *ab initio* DMRG algorithm are not described. To the uninitiated, the discussed algorithm may appear fundamentally different. The problem is exacerbated by the fact that the optimized intermediates in an *ab initio* DMRG program enter in a complicated way in both pure renormalized operator-based and MPO-based implementation. A first goal of this paper is to provide a pedagogical description of how to efficiently implement the *ab initio* DMRG in an MPO-based setting, highlighting the translation to and from the original language of renormalized operators used in renormalized operator-based implementations. This constitutes Section II of this paper.

If the MPO/MPS language only supplied a re-writing of the existing DMRG algorithm, it would be of limited use. However, the power of this language is that certain new perspectives become more natural, and this leads to new algorithms. For example, the MPO/MPS algebra introduces many new operations beyond the scalar computation of a bra-operator-ket expectation value. These operations provide the basis for new algorithms such as DMRG time-evolution.[48–50] One way to implement these algorithms in a pure renormalized operator-based DMRG code would be to augment such codes with explicit MPO's. However, in almost all cases of interest, i.e., if the output of the algorithm is a scalar quantity or an MPS, the operations of the MPO/MPS algebra can be carried out efficiently using only the renormalized operators. Thus, one can build a light-weight layer on top of an existing renormalized operator-based DMRG code to support the relevant MPO/MPS algebra. This strategy is used, for example, in the Block code of some of the authors, to support the MPO/MPS operations needed for perturbation[68–70] and response-based[71,72] calculations on top of DMRG wavefunctions. We here describe this further connection between sweep computations and MPO/MPS algebra in detail in Section III of this work.

MPO/MPS concepts also suggest new formulations of the *ab initio* DMRG sweep algorithm itself. We describe two such formulations here, which can be implemented with equal facility in either a pure renormalized operator-based or MPO-based code. The first is a particular version of the Hamiltonian compression (in a particular MPO gauge) that can be directly applied to the *ab initio* Hamiltonian integrals. This allows for a reduction in the number of renormalized operators built in a DMRG sweep, which can lead to substantial speedups. This algorithm is described in Section IV A, using the linear hydrogen chain as toy computational example. The second is a new way to express the Hamiltonian as a sum of operators, which leads to perfect parallelization of the DMRG algorithm. These ideas are described in Sections IV B and IV C. Finally, our conclusions are provided in Section V.

II. THE DMRG ALGORITHM IN THE MPO AND MPS LANGUAGE

A. The DMRG in renormalization group language

To make the connections clear, we provide a quick refresher of the main concepts of the *ab initio* DMRG sweep algorithm using renormalization group language, as described, for example, in Refs. 4 and 6.

The goal of the DMRG sweep is to compute and/or minimize the energy of the DMRG variational wavefunction. There are variational parameters (renormalized wavefunctions) associated with each of the K orbitals in the problem, thus the sweep consists of iteratively solving a set of ground-state problems, one at a time, associated with each of the K orbitals. To start the procedure, we choose a sequence in which to traverse the orbitals by mapping the orbitals onto the K sites of a one-dimensional lattice. The sweep going from left to right then consists of K steps. At a given step k, we can think of the orbitals as partitioned into two sets, the left block of orbitals L_k and a right block of orbitals R_k, which sets up a tensor product structure of the Hilbert space and operators on the Hilbert space. Associated with the left and right blocks are a set of left and right renormalized states (bases), and left and right renormalized operators. The latter are used as an operator basis to reconstruct the Hamiltonian on all K orbitals, and a proper left-right decomposition of the Hamiltonian (into the so-called normal and complementary operators) is a key step in implementing the *ab initio* DMRG algorithm efficiently.

RIGHTSLINK

For each of the K steps of the sweep, three operations are carried out,

1. blocking, which updates the set of left and right renormalized bases and operators, from the renormalized representations at site $k-1$, to the "blocked" representation at site k;
2. solving, which computes the (ground-state) renormalized wavefunction at site k in the product of the left- and right-renormalized bases,
3. and decimation, which transforms the "blocked" bases and operators to the renormalized representation at site k;

A complete sweep from left to right and back updates all renormalized bases $\{|l_{\alpha_k}\rangle\}$, $\{|r_{\alpha_k}\rangle\}$, and renormalized operators $\{\mathbf{O}^{L_k}_{\beta_k}\}, \{\mathbf{O}^{R_k}_{\beta_k}\}$ for every partition of the orbitals k.

The above operations, and the associated renormalized quantities, are the central objects in the original pure renormalized operator-based *ab initio* DMRG algorithm. *All the same steps and quantities will also appear in an efficient "MPO-based" DMRG implementation*. To make the translation, we thus will be looking to highlight (i) the connection between the renormalized left- and right-bases and renormalized wavefunctions, and the tensors in the MPS, (ii) the correspondence between the left- and right- renormalized operators and the tensors in the Hamiltonian MPO, (iii) the relation between the efficient implementation of DMRG energy minimization and expectation value evaluation with MPS and MPO, and the computational organization into a DMRG sweep algorithm using normal and complementary operators, with the individual steps of blocking, solving, and decimation.

B. Matrix product states

We now recall the basic concepts of MPS. This will also establish some notation useful in discussing MPO's. The relationship between the MPS and the renormalized bases and wavefunctions along a sweep has been discussed before in the chemistry literature, for example, in Refs. 73 and 74. A particularly detailed account is given in Ref. 61.

Matrix product states (MPS) are the wavefunction representations that define the variational space of the DMRG. Within the Fock space of an orthonormal basis of K orbitals, the electronic wavefunction is written in occupation representation as

$$|\Psi\rangle = \sum_{n_1 \cdots n_K} \Psi^{n_1 n_2 \cdots n_K} |n_1 n_2 \cdots n_K\rangle, \tag{1}$$

where $|n_1 n_2 \cdots n_K\rangle$ is an occupancy basis state in the Fock space, and spin-labels have been suppressed. For a given particle number N, we have the condition

$$\Psi^{n_1 n_2 \cdots n_K} = \begin{cases} \Psi^{n_1 n_2 \cdots n_K}, & \sum_{k=1}^{K} n_k = N, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

In a matrix product state, the wavefunction amplitude for a given basis state is written as a product of matrices,

$$\Psi^{n_1 n_2 \cdots n_K} = \sum_{\{\alpha_k\}} A^{n_1}_{\alpha_1}[1] A^{n_2}_{\alpha_1 \alpha_2}[2] \cdots A^{n_K}_{\alpha_{K-1}}[K], \tag{3}$$

where the dimension of $A^{n_k}[k]$ is an $M \times M$ matrix (or a $2 \times M \times M$ tensor if we include the $n_k \in \{0,1\}$ index for spin-orbitals), and the leftmost and rightmost matrices are $1 \times M$ and $M \times 1$ vectors to ensure that the matrix product results in the scalar amplitude $\Psi^{n_1 n_2 \cdots n_K}$. As the dimension M, known variously as the bond-dimension or the number of renormalized states, increases, the representation Eq. (3) becomes increasingly flexible. We will here assume for simplicity that all $A^{n_k}[k]$ are real.

It is very useful to employ a graphical notation for the MPS. In this notation, the general wavefunction amplitude is represented by a tensor with K legs, while the MPS representation is a connected set of 2-index and 3-index tensors, each associated with a site. Contraction between the tensors represents summation, as shown in Fig. 1.

Note that there is a non-uniqueness in the representation since we can always redefine two adjacent matrices

$$A^{n_k}[k] \to A^{n_k}[k] G, \tag{4}$$

$$A^{n_{k+1}}[k+1] \to G^{-1} A^{n_{k+1}}[k+1], \tag{5}$$

where G is an invertible $M \times M$ "gauge" matrix, while leaving the matrix *product* invariant. This redundancy is partially eliminated by placing additional constraints on the matrices, such as the left orthonormality condition $\sum_{n_k} {A^{n_k}}^T A^{n_k} = 1$ and right orthonormality condition $\sum_{n_k} A^{n_k} {A^{n_k}}^T = 1$. Applied to all the tensors, this leads to the left- and right- canonical forms of the MPS, respectively. The DMRG sweep algorithm employs a mixed-canonical form. In this case, at step k of the sweep, all tensors to the left of site k are in left-canonical form, and all tensors to the right of site k are in right-canonical form. The MPS is then expressed as

$$\Psi^{n_1 n_2 \cdots n_K} = \sum_{\{\alpha_k\}} L^{n_1}_{\alpha_1}[1] L^{n_2}_{\alpha_1 \alpha_2}[2] \cdots C^{n_k}_{\alpha_{k-1} \alpha_k}[k] \cdots R^{n_K}_{\alpha_{K-1}}[K], \tag{6}$$

where we have emphasized the choice of gauge by using symbols L, C, R for the different tensors. $C^{n_k}[k]$ is called the DMRG renormalized wavefunction.

The matrices in the MPS define a recursively constructed set of many-body renormalized basis states. These are precisely the left- and right-renormalized bases that are constructed in the DMRG sweep. In this context, the matrices are sometimes called renormalization matrices. For example, if we consider a (bi-)partitioning of the sites at site k and consider the left block of sites $1 \cdots k$, we obtain the left

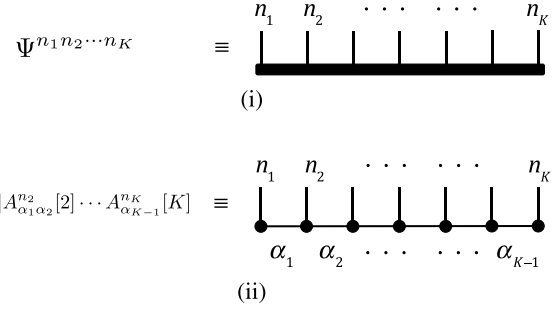

FIG. 1. (i) Wavefunction coefficients in graphical notation. (ii) Representation of wavefunction as a matrix product state in graphical notation (Eq. (3)).

RIGHTSLINK

renormalized basis

$$|l_{\alpha_k}\rangle = \sum_{n_1\cdots n_k} (A^{n_1}[1]A^{n_2}[2]\cdots A^{n_k}[k])_{\alpha_k}|n_1\cdots n_k\rangle \quad (7)$$

and from the right block of sites $k+1 \to K$, we obtain the right renormalized basis

$$|r_{\alpha_k}\rangle = \sum_{n_{k+1}\cdots n_K} (A^{n_{k+1}}[k+1]A^{n_{k+2}}[k+2]\cdots A^{n_K}[K])_{\alpha_k} \times |n_{k+1}\cdots n_K\rangle. \quad (8)$$

The graphical representation of the left and right renormalized basis is shown in Fig. 2. Note that the renormalized states are defined for partitionings at any site k. Iterating through the partitions from $1\cdots K$ builds up the renormalized states in the same recursive fashion as they are built up during a DMRG sweep. In particular, the renormalized states at site $k+1$ are explicitly defined from the renormalized states at site k by the renormalization matrix $A^{n_{k+1}}[k]$, e.g., for the left basis,

$$|l_{\alpha_{k+1}}\rangle = \sum_{\alpha_k n_{k+1}} A^{n_{k+1}}_{\alpha_k\alpha_{k+1}}|l_{\alpha_k}n_{k+1}\rangle \quad (9)$$

and similarly for the right basis. The above transformation Eq. (9) is exactly that of blocking and decimating the states at step $k+1$ of the DMRG sweep: blocking consists of expanding the renormalized basis space, $\{|l_{\alpha_k}\rangle\} \to \{|l_{\alpha_k}n_{k+1}\rangle\}$, while decimation consists of projecting $\{|l_{\alpha_k}n_{k+1}\rangle\} \to \{|l_{\alpha_{k+1}}\rangle\}$.

In determining the tensors $A^{n_k}[k]$ successively in the DMRG sweep, the tensor to be optimized at site k is expressed in the mixed-canonical gauge in Eq. (6) ($C^{n_k}[k]$). In this gauge, the MPS is written in terms of the renormalized states as

$$|\Psi\rangle = \sum_{\alpha_{k-1}n_k\alpha_k} C^{n_k}_{\alpha_{k-1}\alpha_k}|l_{\alpha_{k-1}}n_k r_{\alpha_k}\rangle \quad (10)$$

and thus the coefficients $C^{n_k}_{\alpha_{k-1}\alpha_k}$ are the coefficients of the wavefunction in the DMRG renormalized space. We can also write the MPS more compactly in terms of the left renormalized states at site k, $\{|l_{\alpha_k}\rangle\}$ (rather than the blocked basis $\{|l_{\alpha_{k-1}}n_k\rangle\}$), giving the simpler form

$$|\Psi\rangle = \sum_{\alpha_k} |l_{\alpha_k}r_{\alpha_k}\rangle s_{\alpha_k}. \quad (11)$$

This shows that the MPS corresponds to a wavefunction whose Schmidt decomposition, for the bi-partitioning at any site k, contains at most M singular values s_{α_k}.

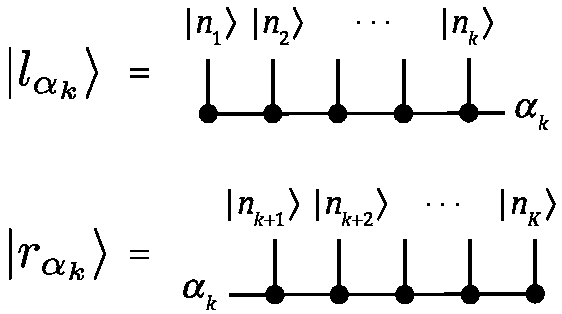

FIG. 2. Left and right renormalized bases at site k (Eqs. (7) and (8)) in graphical notation.

From the above, it is clear that there is no computational distinction to be made between working with the renormalized representations (left, right renormalized bases and renormalized wavefunctions) in a DMRG sweep and the underlying matrix product tensors: since one set is defined in terms of the other, both quantities are always present, in any DMRG implementation, simultaneously.

C. Matrix product operators

We now review the formalism of matrix product operators, emphasizing the similarity with the above analysis for matrix product states. A matrix product operator (MPO) is a generalization of a matrix product representation to the operator space.[44–47] Let us first define an operator basis that spans the operators associated with a given spin-orbital site, such as $\{\hat{z}\} = \{1, a, a^\dagger, a^\dagger a\}$. A general operator can be written as the expansion

$$\hat{O} = \sum_{\{\hat{z}\}} O^{z_1 z_2\cdots z_K}\, \hat{z}_1\hat{z}_2\cdots\hat{z}_K. \quad (12)$$

We introduce a matrix product operator representation as a representation for the element $O^{z_1 z_2\cdots z_K}$

$$O^{z_1 z_2\cdots z_K} = \sum_{\{\beta_k\}} W^{z_1}_{\beta_1}[1]W^{z_2}_{\beta_1\beta_2}[2]\cdots W^{z_K}_{\beta_K}[K]. \quad (13)$$

Note that the entries of $W^{z_k}[k]$ are simply scalars; the operators (which, for example, describe the non-commuting nature of the fermions) are contained within the operator string $\hat{z}_1\hat{z}_2\cdots\hat{z}_K$ in Eq. (12). Also, the decomposition in Eq. (13) is not unique, and contains the same "gauge" redundancy as in the case of the MPS.

It is convenient to define a matrix product operator form where the matrices appearing are operator valued (i.e., the matrix elements are operators). This is done by grouping the operator $\hat{z}_k$ with the corresponding tensor $W^{z_k}[k]$, to define the operator valued matrix $\hat{W}[k]$,

$$\hat{W}_{\beta_{k-1}\beta_k}[k] = \sum_{z_k} W^{z_k}_{\beta_{k-1}\beta_k}[k]\hat{z}_k. \quad (14)$$

The full operator $\hat{O}$ is then a product over the operator valued matrices,

$$\hat{O} = \hat{W}[1]\hat{W}[2]\cdots\hat{W}[K]. \quad (15)$$

An MPO can be expressed in graphical form. Here, it is more conventional to write the operator basis on each site as $\{\hat{z}\} = \{|n\rangle\langle n'|\}$, such that

$$\hat{O} = \sum_{\{n_k n'_k\}} O^{n_1 n_2\cdots n_K}_{n'_1 n'_2\cdots n'_K}|n_1 n_2\cdots n_K\rangle\langle n'_1 n'_2\cdots n'_K|. \quad (16)$$

The MPO representation of the operator matrix element is

$$O^{n_1 n_2\cdots n_K}_{n'_1 n'_2\cdots n'_K} = \sum_{\{\beta_k\}} W^{n_1 n'_1}_{\beta_1}[1]W^{n_2 n'_2}_{\beta_1\beta_2}[2]\cdots W^{n_K n'_K}_{\beta_{K-1}}[K]. \quad (17)$$

A general operator is represented by a tensor with K "up" legs and K "down" legs. The MPO is drawn as a connected set of 3-index and 4-index tensors, each associated with a site, as illustrated in Fig. 3. Note that in this formulation,

RIGHTSLINK

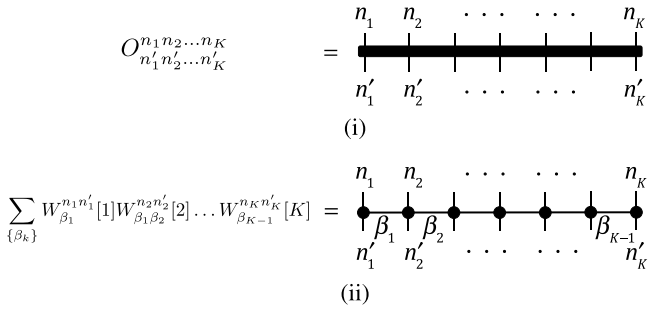

FIG. 3. Matrix product operator (Eq. (16)) in graphical notation.

the non-commuting nature of fermion operators is implicit in the values of the elements of the site-tensors $W^{n_k n'_k}_{\beta_{k-1}\beta_k}[k]$ in Eq. (17).

Similar to the case of MPS, the MPO tensors define sets of many-body operators over a partitioning of the sites. For example, for a partitioning into a left block of sites $1 \cdots k$, we define the corresponding left operators $\hat{O}^L_{\beta_k}$,

$$\hat{O}^L_{\beta_k} = (\hat{W}[1]\hat{W}[2]\cdots\hat{W}[k])_{\beta_k} \tag{18}$$

and from the right block of sites $k+1 \cdots K$, we define a set of right operators, $\hat{O}^R_{\beta_k}$,

$$\hat{O}^R_{\beta_k} = (\hat{W}[k+1]\hat{W}[k+2]\cdots\hat{W}[K])_{\beta_k}. \tag{19}$$

Using the sets of left and right operators, the full operator at any partition can be expressed as

$$\hat{O} = \sum_{\beta_k} \hat{O}^L_{\beta_k}\hat{O}^R_{\beta_k}. \tag{20}$$

Note that the bond-dimension of the MPO at partition k is equal to the number of terms in the summation over β in Eq. (20).

The left-right decomposition of an operator at site k described above is isomorphic to the left-right decomposition of an operator at step k in a DMRG sweep. In particular, the renormalized left-block operators $\mathbf{O}^L_{\beta_k}$ and renormalized right-block operators $\mathbf{O}^R_{\beta_k}$ at step k correspond to projections of $\hat{O}^L_{\beta_k}$, $\hat{O}^R_{\beta_k}$ into the left- and right- renormalized bases,

$$\begin{aligned}[\mathbf{O}^L_{\beta_k}]_{\alpha_k\alpha'_k} &= \langle l_{\alpha_k}|\hat{O}^L_{\beta_k}|l_{\alpha'_k}\rangle,\\ [\mathbf{O}^R_{\beta_k}]_{\alpha_k\alpha'_k} &= \langle r_{\alpha_k}|\hat{O}^R_{\beta_k}|r_{\alpha'_k}\rangle.\end{aligned} \tag{21}$$

These renormalized left- and right-block operators are, of course, the main computational intermediates in a pure renormalized operator-based DMRG implementation, and they play the same role in an MPO-based implementation. The relationship between the left-right decomposition of an operator and the renormalized left-block and right-block operators is shown in graphical form in Fig. 4. We return to their role in efficient computation in Section II E.

The left and right operators at a given partition are explicitly related to the left and right operators at the neighbouring partition. For example, for the left operators, we have

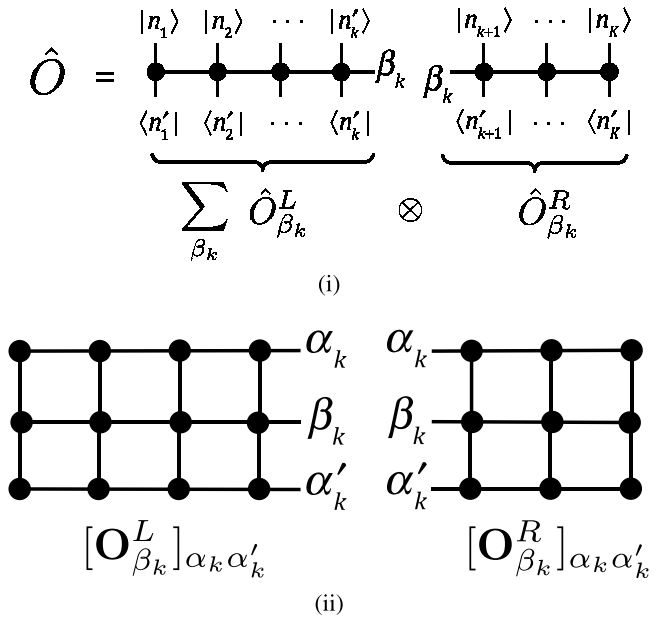

FIG. 4. (i) Left-right decomposition of the MPO at site k. (ii) Left- and right-block renormalized operators at site k.

$$\hat{O}^L_{\beta_k} = \sum_{\beta_{k-1}} \hat{O}^L_{\beta_{k-1}}\hat{W}_{\beta_{k-1}\beta_k}[k], \tag{22}$$

where we can interpret the above as a vector matrix product of the operator valued row-vector $\hat{O}^L$ with the operator valued matrix $\hat{W}[k]$. Analogously for the right operators, we have

$$\hat{O}^R_{\beta_{k-1}} = \sum_{\beta_k} \hat{W}_{\beta_{k-1}\beta_k}[k]\hat{O}^R_{\beta_k}, \tag{23}$$

which can be seen as a matrix vector product. Eqs. (22) and (23) explicitly define the recursion rules that relate the operators for one block of sites, e.g., $1 \cdots k-1$, to a neighbouring block of sites, e.g., $1 \cdots k$. This process of recursively constructing the left- and right-operators at successive blocks is isomorphic to the process of blocking as one proceeds through the sites in a DMRG sweep, the only distinction being that the operators $\hat{O}^R_{\beta_k}$ in Eq. (23) are replaced by their matrix representations $\mathbf{O}^R_{\beta_k}$. We thus refer to the rules as *blocking rules*. As we explain in Section II E, to efficiently compute expectation values we should in fact use the renormalized operators (i.e., operator matrix representations) as in the DMRG sweep, rather than the bare operators themselves, during the blocking process.

It is often convenient for the purposes of interpretation, to write the left-right decomposition of $\hat{O}$ in Eq. (20) a slightly different form,

$$\hat{O} = \hat{O}^{L_k} \otimes \hat{1}^{R_k} + \hat{1}^{L_k} \otimes \hat{O}^{R_k} + \sum_{\beta_k} \hat{o}^{L_k}_{\beta_k}\hat{o}^{R_k}_{\beta_k}. \tag{24}$$

We have introduced 3 kinds of left and right operator terms: the identity operator ($\hat{1}^{L_k}$ or $\hat{1}^{R_k}$), the operator $\hat{O}$ restricted to act on the left or right block of sites ($\hat{O}^{L_k}$ or $\hat{O}^{R_k}$), and terms which express interactions between the left and right sites at partition k ($\hat{o}^{L_k}_{\beta_k}$, $\hat{o}^{R_k}_{\beta_k}$, respectively). Since there are 3 kinds of

RIGHTSLINK

terms, then the matrices and vectors appearing in the blocking rules Eqs. (22) and (23) now have a (3×3) and (3×1) (or (1×3)) block structure, for example, Eq. (23) becomes in expanded form

$$\begin{pmatrix} \hat{O}^{R_k} \\ o^{R_k}_{\beta_k} \\ \hat{1}^{R_k} \end{pmatrix} = \begin{pmatrix} \hat{1}^k & \hat{C}^k & \hat{O}^k \\ 0 & \hat{A}^k & \hat{B}^k \\ 0 & 0 & \hat{1}^k \end{pmatrix} \begin{pmatrix} \hat{O}^{R_{k+1}} \\ \hat{o}^{R_{k+1}}_{\beta_{k+1}} \\ \hat{1}^{R_{k+1}} \end{pmatrix} \tag{25}$$

where the superscript on $\hat{O}^k$ denotes the operator acts on site k.

From the above, we see that building the left-right operator decompositions through the blocking rules in a DMRG sweep is isomorphic to the operations required to construct the explicit MPO; the only difference being that explicit operators are replaced by operator matrices, which is necessary in the efficient computation of expectation values.

D. MPO representation of quantum chemistry Hamiltonians

Based on the efficient left-right decomposition and blocking rules for the *ab initio* Hamiltonian in the standard DMRG algorithm, and the isomorphism to the elements of the MPO tensors $\hat{W}[k]$ established above, we can now easily identify the efficient MPO representation of the quantum chemistry Hamiltonian.

The *ab initio* Hamiltonian is written as

$$\hat{H} = \sum_{pq} t_{pq} a^\dagger_p a_q + \frac{1}{2} \sum_{pqrs} v_{pqrs} a^\dagger_p a^\dagger_q a_r a_s, \tag{26}$$

where spin labels have been suppressed and $v_{pqrs} = \langle pq|sr\rangle = v_{qpsr}$. The summation over the indices is not restricted, thus for a system with K sites, the indices range from $1 \cdots K$.

To obtain the MPO representation, we first identify the left-right decomposition of the Hamiltonian, namely,

$$\hat{H} = \hat{H}^{L_k} \otimes \hat{1}^{R_k} + \hat{1}^{L_k} \otimes \hat{H}^{R_k} + \sum_{\alpha_k} \hat{h}^{L_k}_{\alpha_k} \hat{h}^{R_k}_{\alpha_k}, \tag{27}$$

where the left and right Hamiltonians, are explicitly

$$\hat{H}^{L_k} = \sum_{pq \in L_k} t_{pq} a^\dagger_p a_q + \frac{1}{2} \sum_{pqrs \in L_k} v_{pqrs} a^\dagger_p a^\dagger_q a_r a_s, \tag{28}$$

$$\hat{H}^{R_k} = \sum_{pq \in R_k} t_{pq} a^\dagger_p a_q + \frac{1}{2} \sum_{pqrs \in R_k} v_{pqrs} a^\dagger_p a^\dagger_q a_r a_s, \tag{29}$$

where L_k indicates the domain of indices $1 \cdots k$ (the left block of sites), and R_k the domain of indices $k+1 \cdots K$.

The operators $\hat{h}^{L_k}_{\alpha_k}$ and $\hat{h}^{R_k}_{\alpha_k}$ describe the interactions between the left and right blocks of sites. Although these operators are not uniquely defined (only $\sum_{\alpha_k} \hat{h}^{L_k}_{\alpha_k} \hat{h}^{R_k}_{\alpha_k}$ need remain invariant) the standard *ab initio* DMRG left-right decomposition of the quantum chemistry Hamiltonian provides an efficient and convenient set. In this choice, certain of the operators are associated with electronic integrals (the complementary operators) while other operators are not (the normal operators). Using the notation of Refs. 6 and 7 (see the Appendix of the above references) we can write down a normal/complementary operator decomposition of the Hamiltonian as

$$\begin{aligned} \hat{H} = {} & \hat{H}^{L_k} \otimes \hat{1}^{R_k} + \hat{1}^{L_k} \otimes \hat{H}^{R_k} \\ & + \frac{1}{2} \left(\sum_{p \in L_k} a^\dagger_p \hat{S}^{R_k}_p + h.c. + \sum_{p \in R_k} a^\dagger_p \hat{S}^{L_k}_p + h.c. \right) \\ & + \frac{1}{2} \left(\sum_{pq \in L_k} \hat{A}^{L_k}_{pq} \hat{P}^{R_k}_{pq} + h.c. \right) \\ & - \frac{1}{2} \left(\sum_{pq \in L_k} \hat{B}^{L_k}_{pq} \hat{Q}^{R_k}_{pq} + h.c. \right), \end{aligned} \tag{30}$$

where the various operators are defined as (see also Refs. 6 and 7)

$$\hat{S}^{L_k/R_k}_p = \sum_{q \in L_k/R_k} t_{pq} a_q + \sum_{qrs \in L_k/R_k} w_{pqrs} a^\dagger_q a_r a_s, \tag{31}$$

$$\hat{A}_{pq} = a^\dagger_p a^\dagger_q, \tag{32}$$

$$\hat{B}_{pq} = a^\dagger_p a_q, \tag{33}$$

$$\hat{P}^{R_k}_{pq} = \sum_{rs \in R_k} v_{pqrs} a_r a_s, \tag{34}$$

$$\hat{Q}^{R_k}_{pq} = \sum_{rs \in R_k} \frac{1}{2} x_{prqs} a^\dagger_r a_s = \sum_{rs \in R_k} w_{prqs} a^\dagger_r a_s, \tag{35}$$

with $w_{pqrs} = v_{pqrs} - v_{qprs} = v_{pqrs} - v_{pqsr}$, $x_{pqrs} = v_{pqrs} - v_{qprs} - v_{pqsr} + v_{qpsr} = 2w_{pqrs}$. In the above, the two index complementary operators are chosen to be defined on the right block of sites only. For efficiency, it is possible to use other decompositions where sets of complementary operators are defined on both the left and right blocks. For example, the number of terms in the summation over normal/complementary two index operators will increase during a DMRG sweep as the partition site k is moved from $1 \cdots K$, and the size of the L_k block increases. Thus, for $k > K/2$, efficient DMRG sweep implementations switch to a representation where the two index complementary operators are chosen to be defined on the left block of sites. In addition, fermionic symmetries (such as $B_{pq} = -B^\dagger_{qp}$ for $p > q$) are used.

From the double summation over pq, it is clear that the number of terms appearing in Eq. (30) is $O(K^2)$, thus the total bond-dimension of the MPO representation of the Hamiltonian is also $O(K^2)$. The prefactor in the $O(K^2)$ bond-dimension depends on the particular choice of splitting between normal and complementary operators, and how the integrals are distributed. Several cases are worked out explicitly in the Appendix. For example, Fig. 10 shows explicitly that the bond-dimension is minimized by using the switch between left and right complementary operators at the middle site $k = K/2$, as discussed above.

As we have explained, the $\hat{W}[k]$ matrix of the MPO encodes the blocking rule that takes the left/right operators at one partitioning to a neighbouring partitioning. For the choice of normal/complementary operators in Eq. (35), the blocking rules can be found in the original DMRG quantum chemistry algorithm descriptions, see, e.g., Eqs. (A1)-(A10)

RIGHTSLINK

in the Appendix of Ref. 7, and from these rules we can read off the $\hat{A}, \hat{B}, \hat{C}$ in Eq. (25). For example, the rule to construct the operator $\hat{P}^{R_k}_{pq}$ from the operators in the previous partition is given in Eq. (A7) in Ref. 7,

$$\hat{P}^{R_k}_{pq} = \hat{1}^k \otimes \hat{P}^{R_{k+1}}_{pq} + \hat{P}^k_{pq} \otimes \hat{1}^{R_{k+1}} + \sum_{s \in R_{k+1}} w_{pqks} a_k \otimes a_s, \tag{36}$$

where we have used the fact that the additional site relating R_k and R_{k+1} has orbital index k, and $\hat{1}^k$, $\hat{P}^k_{pq}$, a_k denote the corresponding operators defined on site k. The blocking rule (36) corresponds to a matrix vector product in Eq. (25),

$$\begin{pmatrix} \vdots \\ \hat{P}^{R_k}_{pq} \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \cdots & \hat{A}_0 & \hat{A}_1 & \hat{B} \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \vdots \\ a_n \\ \hat{P}^{R_{k+1}}_{pq} \\ 1 \end{pmatrix}, \tag{37}$$

with the correspondence $\hat{A} = (\hat{A}_0, \hat{A}_1)$, $\hat{o}^{R_{k+1}}_{\alpha_{k+1}} = (a_n, \hat{P}^{R_{k+1}}_{pq})$. $\hat{A}_0$ has elements $[\hat{A}_0]_{pq,s} = v_{pqks} a_k$, $\hat{A}_1$ is an identity matrix, and $\hat{B}$ is a diagonal matrix with diagonal entries $\hat{P}^k_{pq}$. The blocking rule of the operators used in the original DMRG algorithm thus explicitly carries out the matrix-vector multiplication of $\hat{W}[k]$ in the MPO in an element-wise notation.

Finally, we note that the approach used by Keller *et al.* in Ref. 64, the so-called *fork-fork-merge* or *fork-merge-merge* operations for reusing common intermediate operators, is completely equivalent to using $\hat{P}$ and $\hat{Q}$ complementary operators in the *right* or *left* block, respectively. Specifically, in Figures 2 and 3 of Ref. 64, two-electron integrals with two indices on the left, one index on site k, and one index on the right, are all collected into the MPO matrix $\hat{W}[k]$, similarly to Eq. (37).

E. Efficient implementation of expectation values

We have so far established the correspondence between the language of MPO/MPS and the renormalized states and operators used in a pure renormalized operator-based DMRG implementation. We now discuss how to efficiently compute expectation values (such as the energy) in an MPO-based DMRG implementation. Two questions arise: how to use the structure and sparsity of the MPO tensors, and the order in which to perform contractions between the MPO and MPS. In fact, both aspects are addressed by the original DMRG sweep algorithm, by using element-wise blocking operations, separate blocking and decimation steps, and by building the renormalized operators mentioned in Sec. II C as computational intermediates. We now discuss how these individual components arise in an MPO-based expectation value computation.

To see why MPO tensor sparsity is important, we first observe that the cost of the quantum chemistry DMRG sweep algorithm to compute (or minimize) the energy is $O(K^4)$, which is what one would expect given that the Hamiltonian contains $O(K^4)$ fermionic terms. However, if we try to reconstruct the Hamiltonian operator $\hat{H}$ from its $\hat{W}[k]$ MPO product (15) using dense matrix algebra, we formally require $O(K^5)$ operations, as first noted in Ref. 62. This is because multiplying out Eq. (15) requires $O(K)$ matrix vector products between the $O(K^2) \times O(K^2)$ dimension $\hat{W}[k]$ matrices, and the $K^2 \times 1$ boundary $\hat{W}$ vectors ($\hat{W}[1]$ or $\hat{W}[K]$).

The reason for the incorrect scaling of $O(K^5)$ is that the above argument neglects the fact that $\hat{W}$ contains many zero elements.[62] To see this explicitly, we consider Eq. (37) that determines the update of the $\hat{P}_{pq}$ elements of $\hat{W}$. Here, the multiplication of the $\hat{A}_1$ and $\hat{B}$ matrices into the column vector formally takes $O(K^4)$ cost, which repeated over the $O(K)$ $\hat{W}[k]$ matrices leads to the incorrect $O(K^5)$ scaling. However, the $\hat{A}_1$ and $\hat{B}$ matrices are in fact *diagonal* matrices and can be multiplied with $O(K^3)$ cost over all the $\hat{W}$ matrices. The main cost in Eq. (37) arises then from the multiplication of the $\hat{A}_0$ matrix (of dimension $O(K^2 \times K)$) into the $O(K)$ a_s operators. This is of $O(K^3)$ cost for a single multiplication, and of $O(K^4)$ cost over all the $\hat{W}$ matrices, leading to the correct scaling.

Note also that there are many symmetries between different elements of $\hat{W}$. For example, although both $a^\dagger_p$, a_p appear as elements of $\hat{W}$, they are related by the Hermitian conjugation (and similarly for elements such as $a^\dagger_p a^\dagger_q$, $a_p a_q$ and the $p > q$ and $p < q$ components of $a^\dagger_p a_q$). These elements would be manipulated and multiplied separately in a simple implementation of an MPO. However, such symmetries and relationships can further be used to reduce the prefactor of the reconstruction of $\hat{H}$ as well as the storage of the MPO's.

The explicit expressions for blocking in the original DMRG algorithm are *element-wise* expressions of the multiplications of $\hat{W}$ which already incorporate both the sparsity and symmetry between the elements, and thus lead to efficient operations with $\hat{H}$. To efficiently carry out blocking in an MPO-based implementation, the same element-wise strategy should be used. This can be achieved in practice by storing additional meta-information on the non-zero matrix elements and how to multiply them, as is done, for example, in MPSXX[62] and QC-Maquis.[64]

We now consider contracting the Hamiltonian MPO with the bra and ket MPS to compute the energy, $E = \langle \Psi | \hat{H} | \Psi \rangle$. As $|\Psi\rangle$ is an MPS and $\hat{H}$ is an MPO, we could imagine first computing $\hat{H}|\Psi\rangle$ (obtaining a new MPS) before contracting with the bra. However, it is easy to see that this leads again to the wrong scaling, because the intermediate $\hat{H}|\Psi\rangle$ is now an MPS of very large bond dimension $O(MK^2)$, requiring a very large amount of storage. Instead, one should contract the tensors of the MPS bra and ket with the tensors of the MPO, site by site from $1 \cdots K$. *This corresponds exactly to the recursive construction of the renormalized operators through blocking and decimation along a sweep* (see Fig. 5).

To illustrate how this recursive construction arises naturally, we first define a partial expectation value over a site k as the matrix $E[k]$ (sometimes called the transfer operator),

$$E[k]_{\gamma_{k-1}, \gamma_k} = \sum_{n_k n'_k} \langle n_k | A^{n_k}_{\alpha_{k-1} \alpha_k} \hat{W}_{\beta_{k-1}, \beta_k}[k] A^{n'_k}_{\alpha'_{k-1} \alpha'_k} | n'_k \rangle, \tag{38}$$

RIGHTSLINK

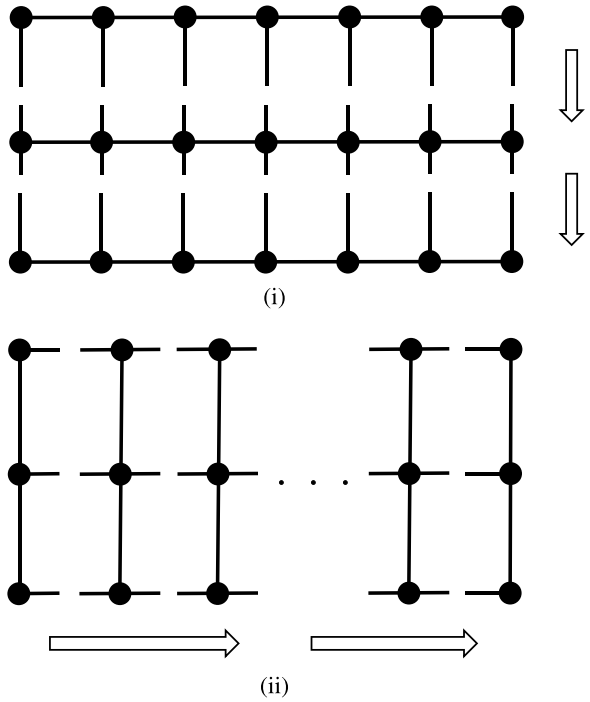

FIG. 5. (i) Incorrect contraction order for an expectation value. (ii) Contraction order, leading to renormalized operators, for an expectation value. The individual tensors appearing correspond to the transfer operators in Eq. (38).

where the compound index $(\gamma_{k-1},\gamma_k) \equiv (\alpha_{k-1}\alpha'_{k-1}\beta_{k-1}, \alpha_k\alpha'_k\beta_k)$. The energy expectation value can be written as

$$E = E[1]E[2]\cdots E[K], \tag{39}$$

where $E[k]$ is an $O(M^2K^2)\times O(M^2K^2)$ matrix, and $E[1]$ and $E[K]$ are $1\times O(M^2K^2)$ and $O(M^2K^2)\times 1$ vectors. Graphically, we illustrate the energy expectation computation by Fig. 5.

When carrying out the energy computation, one naturally multiplies the matrices together from the left or from the right. Multiplying up to site k from the left or the right, respectively, defines the left and right operator matrix representations, namely

$$\begin{aligned}[\mathbf{O}^L_{\beta_k}]_{\alpha_k\alpha'_k} &= (E[1]E[2]\cdots E[k])_{\gamma_k},\\ [\mathbf{O}^R_{\beta_k}]_{\alpha_k\alpha'_k} &= (E[k+1]E[k+2]\cdots E[K])_{\gamma_k},\end{aligned} \tag{40}$$

where α_k,α'_k denote the matrix indices of the renormalized operator matrices, and the different renormalized operators are indexed by β_k (cf. Eq. (21)). $\mathbf{O}^L_{\beta_k}$ and $\mathbf{O}^R_{\beta_k}$ are of course the same left- and right-renormalized operators that appear in the left-right decomposition Hamiltonian at site k and are the standard intermediates in a DMRG sweep.

What is the cost to build the renormalized operators? A naive multiplication of the K $E[k]$ matrices is a multiplication of $O(M^2K^2)\times O(M^2K^2)$ matrices into an $O(M^2K^2)$ length vector. Carrying this out $O(K)$ times would appear to require $O(M^4K^5)$ cost, which is higher than cost of the *ab initio* DMRG algorithm. However, in a standard DMRG sweep implementation (cf. Section II), the renormalized operators are built in two steps: first blocking, then decimation. This is equivalent to observing that $E[k]$ is itself composed of a tensor contraction, and thus we can perform multiplication of two $E[k]$ matrices in two smaller steps (Fig. 6). This reduces the cost of multiplying the K $E[k]$ matrices (and building the renormalized operators) to $O(M^3K^3) + O(M^2K^5)$. This is the lowest cost if we assume that the $E[k]$ matrices are dense and is the generic cost associated with evaluating the expectation value of an MPO with bond dimension $O(K^2)$ with an MPS of bond dimension $O(M^2)$. However, the high $O(K^5)$ scaling is once again (as noted in Ref. 62) because we have not yet accounted for the sparsity of the $E[k]$ matrices. By using elementwise blocking rules (as in Eqs. (41) and (42)), we can explicitly carry out the elementwise multiplication of the $E[k]$ matrices taking into account the appropriate sparsity, as well as the symmetries of the elements of $E[k]$. For example, the blocking operation followed by decimation, for the $\mathbf{P}_{pq}$ element of the $\mathbf{O}^{R_k}$ corresponds to (blocking)

$$\mathbf{P}^{R_k}_{pq} = \mathbf{1}^k \otimes \mathbf{P}^{R_{k+1}}_{pq} + \mathbf{P}^k_{pq} \otimes \mathbf{1}^{R_{k+1}} + \sum_{n\in R_{k+1}} w_{pqks}\, \mathbf{a}_k \otimes \mathbf{a}_s \tag{41}$$

followed by (decimation)

$$[\mathbf{P}^{R_k}_{pq}]_{\alpha_k,\alpha'_k} \leftarrow \sum_{n_k\alpha_k,n'_k\alpha'_k} A^{n_k}_{\alpha_k\alpha_{k+1}} [\mathbf{P}^{R_k}_{pq}]_{n_k\alpha_{k+1},n'_k\alpha_{k+1}} A^{n'_k}_{\alpha'_k\alpha'_{k+1}}. \tag{42}$$

Incorporating elementwise blocking and decimation steps then leads finally to the correct cost of $O(M^3K^3) + O(M^2K^4)$ (the cost of the original DMRG quantum chemistry algorithm). In summary, this allows an MPO-based implementation of

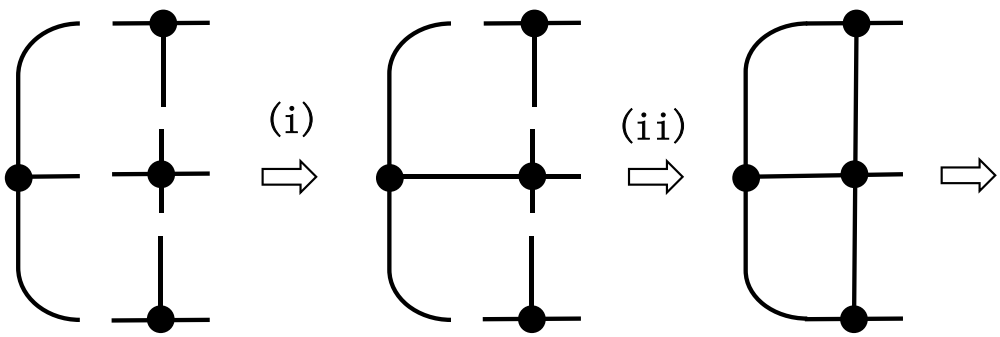

FIG. 6. Individual steps in an expectation value contraction. (i) corresponds to renormalized operator blocking, while (ii) corresponds to renormalized operator decimation.

RIGHTSLINK

DMRG to recover the same cost as a pure renormalized operator-based implementation, through essentially an identical set of computations.

III. MPO AND MPS ALGEBRA IN A RENORMALIZED OPERATOR-BASED IMPLEMENTATION

In Sec. II, we focused on the relationship between the efficient computation of expectation values within an MPO-based DMRG implementation, and the same computation within a pure renormalized operator-based implementation. We saw that a natural way to achieve the same scaling in an MPO-based implementation is to map the computations in the standard DMRG sweep to the MPO-based language.

Expectation values are the natural target of the DMRG sweep algorithm. The algebra of matrix product operators and matrix product states extends beyond expectation values, however, and many more general MPO-MPS operations appear in a variety of algorithmic contexts. For example, to time-evolve an MPS with maximum bond-dimension M, involves repeating the following sequence of operations, for each time step:[48–50]

1. $|\Psi(t)[M]\rangle \rightarrow e^{-i\epsilon\hat{H}}|\Psi(t)[M]\rangle \equiv |\Psi(t+\epsilon)[M']\rangle$ (evolution),
2. $|\Psi(t+\epsilon)[M']\rangle \rightarrow |\Psi(t+\epsilon)[M]\rangle$ (compression).

An important question is whether or not this kind of algorithm, involving a more general MPO/MPS algebra, can be supported within a pure renormalized operator-based DMRG implementation, where only the renormalized operators appear. The answer is that *any* MPO/MPS operation, whose final result is a scalar or an MPS, can in fact be easily implemented within a pure-sweep implementation without any major effort. Consider, for example, the time-evolution operation above. The first step is an MPO × MPS product, which is not part of the standard DMRG sweep. However, the combination of the two steps (including the compression) *is* in the form of a sweep computation, since compression corresponds to maximizing the overlap (i.e., expectation value) $\langle\Phi[M]|e^{-i\epsilon\hat{H}}|\Psi[M]\rangle$ with respect to $\langle\Phi|$. In fact, one can even obtain the full MPS $e^{-i\epsilon\hat{H}}|\Psi[M]\rangle$ with no compression, by simply requiring, in the overlap maximization sweep, that the bond dimension of $\langle\Phi|$ is kept as $M \times D$, where D is the bond dimension of the MPO $e^{-i\epsilon\hat{H}}$ (and thus no compression occurs).

To compute the action of a product of matrix product operators on a matrix product state, one simply has to apply the above procedure multiple times. For example, to obtain $\langle\Psi|\hat{O}\hat{O}|\Psi\rangle$, we first maximize the overlap $\langle\Phi|\hat{O}|\Psi\rangle$ to determine $\langle\Phi|$ and then compute the overlap $\langle\Psi|\hat{O}|\Phi\rangle$.

Only algorithms for whom the final output is an MPO itself (which is rare in zero-temperature calculations) require a full implementation of MPO functionality beyond renormalized operator computation. Implementing the general MPO/MPS algebra as described above can be achieved by updating a renormalized operator-based DMRG code with a simple interface. This is what is found, for example, in the MPO/MPS implementation within the Block code, as is used in DMRG response[71,72] and perturbation calculations.[68–70]

IV. IMPROVING DMRG THROUGH MATRIX PRODUCT OPERATORS

A. Hamiltonian compression

In this section, we focus on some of the new ideas brought by matrix product operators to the implementation of DMRG-like algorithms.

The simplest observation is that, in the same way that it is possible to compress an MPS, it is also possible to compress an MPO. Consequently, in all algorithms where, for example, an operator appears, it is possible to carry out an approximate computation using a compressed version of the same operator. In some cases, this can lead to very substantial savings. For example, for two-point interactions that are a sum of D exponentials, such as $\sum_{ij} V_{ij} n_i n_j$ where $V_{ij} = \sum_\lambda \exp(\lambda|i-j|)$ then the MPO can be compressed exactly to have bond dimension D. This means that, for example, when carrying out a DMRG calculation in a one-dimension system using a short-ranged (e.g., sum of exponentials) interaction, it is possible to carry out such a calculation with a cost that is linear with the length of the system.

In general, a unique compression scheme in an MPO requires choosing a gauge convention. A particularly simple way to arrange the compression is to start from a left-right decomposition of the Hamiltonian at each site i,

$$\hat{H} = \hat{H}^{L_k} \otimes \hat{1}^{R_k} + \hat{1}^{L_k} \otimes \hat{H}^{R_k} + \sum_{\alpha_k,\beta_k} h_{\alpha_k,\beta_k} \hat{h}^{L_k}_{\alpha_k} \hat{h}^{R_k}_{\beta_k}. \tag{43}$$

In Eq. (43) the operators $\hat{h}^{L_k}$ and $\hat{h}^{R_k}$ are purely fermionic operators (normal operators) and do not have any one- or two-particle integrals attached; the corresponding one- and two-particle integrals are stored in the matrix h_{α_k,β_k}. For example, considering only the one-particle part of the Hamiltonian, the interaction term in Eq. (43) would become

$$\sum_{\alpha_k,\beta_k} h_{\alpha_k,\beta_k} \hat{h}^{L_k}_{\alpha_k} \hat{h}^{R_k}_{\beta_k} \rightarrow \sum_{p\in L_k, q\in R_k} t_{pq}(a^\dagger_p a_q + h.c.). \tag{44}$$

We can then compress the MPO by simply considering the singular value decomposition of the matrix h_{α_k,β_k}, $h = U\lambda V^\dagger$, defining the left and right operators as $\hat{h}^{L_k}U$ and $V^\dagger\hat{h}^{R_k}$, and dropping small singular values. (Note that due to quantum number symmetries, h_{α_k,β_k} is block diagonal, thus the singular value decomposition can be carried out on the separate blocks.)

The left-right decomposition of the Hamiltonian becomes

$$\hat{H} = \hat{H}^{L_k} \otimes \hat{1}^{R_k} + \hat{1}^{L_k} \otimes \hat{H}^{R_k} + \sum_i \lambda_i \left(\sum_{\alpha_k} \hat{h}^{L_k}_{\alpha_k} U_{\alpha_k i}\right)\left(\sum_{\beta_k} V^\dagger_{i\beta_k} \hat{h}^{R_k}_{\beta_k}\right), \tag{45}$$

and the corresponding transformation of the $\hat{W}[k]$ matrices appearing in Eq. (25) is

$$\hat{O}^k \rightarrow \hat{O}^k, \tag{46}$$

$$\hat{A} \rightarrow V^\dagger_{k-1}\hat{A}V_k, \tag{47}$$

RIGHTSLINK

$$\hat{B} \to V_{k-1}^{\dagger} \hat{B}, \tag{48}$$

$$\hat{C} \to \hat{C} V_k. \tag{49}$$

Note that the left-right decomposition has the same summation structure as in the standard DMRG representation, only the number of indices summed over is smaller, since small singular values λ_i are dropped. Consequently, standard strategies for parallelization in DMRG which involve parallelizing over the left-right decomposition sum (see Sec. IV C) may be used without modification with the compressed representation.

To illustrate this compression in an *ab initio* quantum chemistry context, we have implemented the above scheme to compute the variational energy of a linear chain of 20 equally spaced hydrogen atoms in the minimal STO-3G basis. Shown in Fig. 7 is the exact energy versus bond-length curve computer using a DMRG calculation with $M = 1000$. Also shown are the errors of using an approximate compressed MPO, with the error shown versus the truncation threshold of the MPO (Fig. 8), as well as the bond-dimension of the MPO (Fig. 9), for spacing $R = 1.0$ Å, 2.0 Å, 3.6 Å. We see that the error in the energy is proportional to the truncation threshold, and exponentially decreases with the bond-dimension of the MPO. Note that the full bond-dimension of the MPO in our choice of gauge in this system varies from 43 084 (bond-length of 1.0 Å) to 15 096 (bond-length of 3.6 Å). However, to obtain an error of 10^{-6} E_h, it is sufficient to use an MPO bond-dimension less than 200. Given that the cost of each step in the DMRG sweep is proportional to the bond-dimension of the MPO, this is a factor of 100 in savings.

B. Efficient sum of operators representation

In Section II E we saw that a naive implementation of DMRG using an MPO representation with dense matrices leads to an incorrect scaling algorithm and that the standard *ab initio* DMRG algorithm corresponds to encoding the sparse matrix multiplications of the MPO to obtain an optimal scaling.

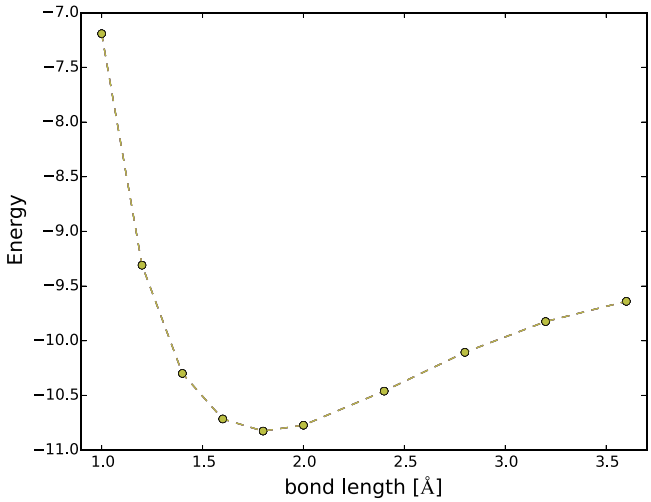

FIG. 7. Exact energy versus bond-length for the symmetric stretch of a chain of 20 hydrogen atoms.

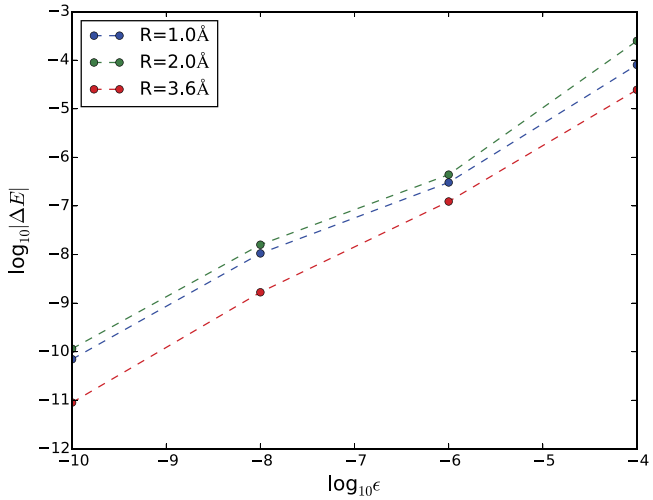

FIG. 8. Total energy error corresponding to a given singular value truncation threshold at bond-lengths 1.0 Å, 2.0 Å, and 3.6 Å.

There is, however, a different and quite simple way to formulate an MPO representation which, even when using naive dense matrix algebra, recovers the correct $O(K^4)$ scaling in a quantum chemistry algorithm. This is achieved by abandoning a single MPO expression for the Hamiltonian, and instead rewriting the Hamiltonian as a sum of sub-Hamiltonians $\hat{H}_m$, where each term is separately represented by an MPO. Each sub-Hamiltonian $\hat{H}_m$ in Eq. (51) is defined as a Hamiltonian where the integrals have a restriction on the first one- or two-electron integral index,

$$\hat{H} = \sum_m \hat{H}_m, \tag{50}$$

$$\hat{H}_m = \sum_q t_{mq} a_m^{\dagger} a_q + \frac{1}{2} \sum_{qrs} v_{mqrs} a_m^{\dagger} a_q^{\dagger} a_r a_s. \tag{51}$$

The MPO representation of $\hat{H}_m$ has bond-dimension $O(K)$. We can see this by once again working with the left-right decomposition, writing

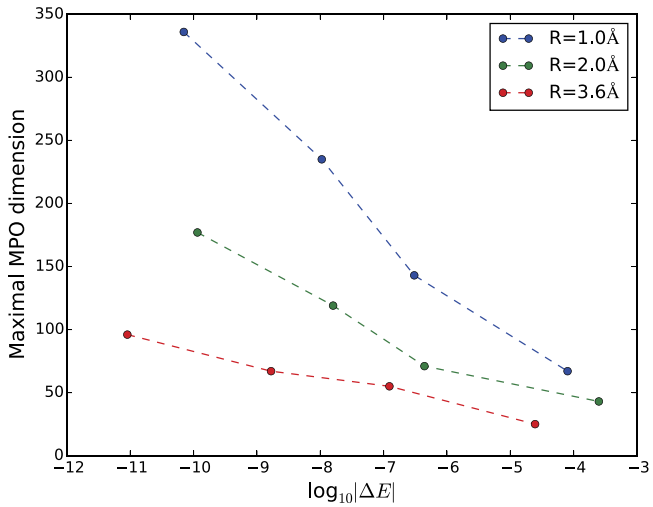

FIG. 9. Bond dimension corresponding to a given total energy error at bond-lengths 1.0 Å, 2.0 Å, and 3.6 Å.

RIGHTSLINK

$$\hat{H}_m = a_m^\dagger \hat{T}_m, \tag{52}$$

$$\begin{aligned}\hat{T}_m &= \sum_q t_{mq} a_q + \frac{1}{2}\sum_{qrs} v_{mqrs} a_q^\dagger a_r a_s \\ &= \hat{T}_m^{L_k} \otimes \hat{1}^{R_k} + \hat{1}^{L_k} \otimes \hat{T}_m^{R_k} \\ &\quad + \frac{1}{2}\sum_{q\in L_k} [a_q^\dagger \hat{P}_{mq}^{R_k} - a_q \hat{Q}_{mq}^{R_k}] \\ &\quad + \frac{1}{2}\sum_{q\in R_k} [\hat{P}_{mq}^{L_k} a_q^\dagger - \hat{Q}_{mq}^{L_k} a_q].\end{aligned} \tag{53, 54}$$

T_m is a sum over $O(K)$ terms and thus has bond-dimension $O(K)$. Since $a_m^\dagger$ is an MPO of bond-dimension 1, $\hat{H}_m$ (as a product of $a_m^\dagger$ and T_m) is also of bond-dimension $O(K)$.

Note that the above is not the only way to split $\hat{H}$ into sub-Hamiltonians. For example, $\hat{H}_m$ could alternatively be defined as a collection of terms sharing the same rightmost operator on the one-dimensional orbital lattice. The same scaling of the bond-dimension $O(K)$ is obtained, but the bond dimensions to the right of the site m then become simply 1. This lowers the average bond-dimension of the MPO across the lattice. (For details, see the Appendix.)

An immediate consequence of rewriting the Hamiltonian as a sum over the K MPO operators $\hat{H}_m$ of bond-dimension $O(K)$ is that the naive (dense matrix algebra) cost of working with the MPO retains the *correct* $O(K^4)$ scaling of quantum chemistry algorithms. For example, consider the reconstruction of $\hat{H}$ from its $\hat{W}$ matrix decomposition Eq. (15). For each $\hat{H}_m$ we have

$$\hat{H}_m = \hat{W}_m[1]\hat{W}_m[2]\cdots\hat{W}_m[K], \tag{55}$$

where each $\hat{W}_m[k]$ matrix is an $O(K)\times O(K)$ matrix. Even if we manipulate each $\hat{W}_m[k]$ matrix as a dense matrix, the cost of multiplying out the terms in Eq. (55) is $O(K^3)$ for each $\hat{H}_m$, and thus $O(K^4)$ cost when considering all K $\hat{H}_m$ operators. This is the correct physical scaling as contrasted with the $O(K^5)$ scaling with the naive MPO representation algorithm. In a similar fashion, the cost to evaluate the energy expectation value in the sum of Hamiltonians representation is $O(M^3K^3) + O(M^2K^4)$ i.e., the same scaling as the quantum chemistry DMRG algorithm.

The decomposition of the Hamiltonian into $\hat{H}_m$ can be seen as a way of using the inherent sparsity in the MPO representation of $\hat{H}$, to recover the correct scaling. However, although the correct scaling is achieved even when using dense matrix algebra in this representation, the prefactor is significantly larger than the standard *ab initio* DMRG algorithm, if we do not use additional sparsity in the $\hat{W}_m$ matrices. Consider, for example, the renormalization rule for $\hat{P}_{mq}^{R_k}$ in Eq. (54), given by Eq. (37). Here, since there are only $O(K)$ $\hat{P}_{mq}^{R_k}$ operators, the $\hat{A}_1$ matrix is an $O(K)\times O(K)$ identity matrix. The dense multiplication of this matrix is only of $O(K^3)$ cost and leads to a physically correct $O(K^4)$ scaling when all $\hat{W}_m$ matrices are multiplied over. However, it is clear that without taking into account the zeros in the identity matrix, we will still perform too many operations.

C. Perfect parallelism in the sum of operators formulation

Parallelization is a key component of practical DMRG calculations in quantum chemistry. There are three principal sources of parallelism that have been so far been considered in DMRG calculations:[7,75–79] (i) parallelism over the left-right decomposition of the Hamiltonian,[7] (ii) parallelism over "quantum numbers" in the DMRG renormalized operator matrices,[76] and (iii) parallelism over sites in the sweep algorithm.[78] Out of these three sources, only (i) and (ii) have been actually implemented in the context of quantum chemistry. The sources of parallelism are largely independent and can be combined to give multiplicative speed up in a parallel DMRG implementation and utilized in modern implementations.

For typical systems, the largest source of parallelism is source (i), i.e., the left-right decomposition. In this case parallelism is expressed over the loop over the normal and complementary operators appearing in Eq. (27), i.e.,

$$\sum_{\alpha_k} \hat{h}_{\alpha_k}^{L_k} \hat{h}_{\alpha_k}^{R_k} \rightarrow \sum_{\text{proc}} \sum_{\alpha_k \in \text{proc}} \hat{h}_{\alpha_k}^{L_k} \hat{h}_{\alpha_k}^{R_k}, \tag{56}$$

where different $\hat{h}_{\alpha_k}^{L_k}, \hat{h}_{\alpha_k}^{R_k}$ are stored and manipulated on different cores/processors. This is an efficient source of parallelism because there are $O(K^2)$ terms in the sum, thus even for a modest number of orbitals (e.g., $K = 50$) it is possible to distribute operations over a large number cores. However, there still remain important communication steps, as the renormalization rules (see, e.g., Eq. (36) and Eqs. (A1)-(A10) in Ref. 7) to build the different normal and complementary operators defined in Eq. (35), involve several different kinds of normal and complementary operators. For example, in Eq. (36), to construct $P_{ij}^{R_k}$ we need not only $P_{ij}^{R_{k+1}}$ but also the identity operator (which is trivial), as well as a_k and a_n operator matrices. If the a_n operators are not stored on the processor that also stores $P_{ij}^{R_k}$, then it must be communicated.

An important advantage of the sum over operators formulation in Section IV B is that each sub-Hamiltonian term $\hat{H}_m$ can be manipulated completely independently of any other term. Thus the construction of $\hat{H}_m$, the associated renormalized operators, and renormalized operator matrices for each $\hat{H}_m$, can be carried out independently of every other $\hat{H}_m$. This leads to a different organization of the parallelization of the DMRG algorithm, which is highly scalable up to $O(K)$ processes. Compared to the most common parallelization strategy (i) there is no need to communicate renormalized operators between processes; only the renormalized wavefunction need be communicated, leading to a substantial decrease in communication cost, while the leading order memory and computation requirements remain unaffected. Further, for each sub-Hamiltonian, one may further parallelize its operations through strategies (i), (ii), and (iii) above. The investigation of the scalability of the promising sum over operator parallelization is thus of interest in future work.

RIGHTSLINK

V. CONCLUSIONS

In this work we had three goals, namely, (i) to explain how to efficiently implement the *ab initio* DMRG in the language of matrix product operators and matrix product states, in particular, highlighting the connection to the original description of the *ab initio* DMRG sweep algorithm, (ii) to discuss the implementation of more general matrix product operator/matrix product state algebra within the context of a DMRG sweep with renormalized operators, and (iii) to describe some ways that thinking about matrix product operators can lead to new formulations of the DMRG, using compression and parallelism as examples.

In recent years, many extensions of the *ab initio* DMRG have appeared which are motivated by the very convenient matrix product operator/matrix product state formalism. As these developments continue, the connections established in this work provide a bridge to translate these conceptual advances into efficient implementations, using the long-standing technology of the DMRG.

ACKNOWLEDGMENTS

G. K.-L. Chan would like to acknowledge the US Department of Energy for funding primarily through No. DE-SC0010530, with additional funding provided by No. DE-SC0008624. Z. Li was supported by the Simons Foundation through the Simons Collaboration on the Many-Electron problem. S. R. White would like to acknowledge funding from the Simons Foundation through the Simons Collaboration on the Many-Electron problem.

APPENDIX: ANALYSIS OF THE DISTRIBUTIONS OF BOND DIMENSIONS FOR DIFFERENT CHOICES OF INTERMEDIATES

As discussed in the main text, for two-body Hamiltonians there is freedom to choose different normal and complementary operator intermediates, all of which result in the same scaling of $O(K^2)$ for the bond dimension. Here, we analyze in more detail how different choices of intermediates lead to different distributions of the leading bond dimensions $D(k)$ along the one-dimensional array of orbital partitions, $k \in \{1,\ldots,K-1\}$. Only the two body terms in Eq. (26) will be examined, as the inclusion of the one-body term does not change the leading bond dimensions $D(k)$. To further simplify the discussion, we use the following form of two-electron integrals, viz., $\hat{H}_2 = \frac{1}{2} v_{pqrs} a_p^\dagger a_q^\dagger a_r a_s = g_{pqrs} a_p^\dagger a_q^\dagger a_r a_s$, where the Einstein summation convention for repeated indices has been assumed, and the tensor g_{pqrs} represents the unique two-electron integrals, whose number is of $O(K^4/4)$,

$$g_{pqrs} = \begin{cases} w_{pqrs}, & p<q,\ r<s \\ 0, & \text{otherwise} \end{cases}. \tag{A1}$$

To examine $D(k)$ for $\hat{H}_2$, we consider the left-right bipartition of orbitals, in which case $\hat{H}_2$ can be written as

$$\begin{aligned}\hat{H}_2 = \hat{H}_2^L + \hat{H}_2^R &- g_{p_L q_R r_L s_R}(a_{p_L}^\dagger a_{r_L})(a_{q_R}^\dagger a_{s_R}) + g_{p_L q_L r_R s_R}(a_{p_L}^\dagger a_{q_L}^\dagger)(a_{r_R} a_{s_R}) + g_{p_R q_R r_L s_L}(a_{r_L} a_{s_L})(a_{p_R}^\dagger a_{q_R}^\dagger) \\ &+ g_{p_L q_R r_R s_R} a_{p_L}^\dagger (a_{q_R}^\dagger a_{r_R} a_{s_R}) + g_{p_L q_R r_L s_L}(a_{p_L}^\dagger a_{r_L} a_{s_L}) a_{q_R}^\dagger \\ &+ g_{p_R q_R r_L s_R} a_{r_L}(a_{p_R}^\dagger a_{q_R}^\dagger a_{s_R}) + g_{p_L q_L r_L s_R}(a_{p_L}^\dagger a_{q_L}^\dagger a_{r_L}) a_{s_R}. \end{aligned} \tag{A2}$$

To minimize the bond dimensions across the left and right blocks, the unambiguous choice is to define the following intermediates for the last four terms:

$$(\hat{T}1)_{p_L}^R \triangleq g_{p_L q_R r_R s_R}(a_{q_R}^\dagger a_{r_R} a_{s_R}), \tag{A3}$$

$$(\hat{T}2)_{q_R}^L \triangleq g_{p_L q_R r_L s_L}(a_{p_L}^\dagger a_{r_L} a_{s_L}), \tag{A4}$$

$$(\hat{T}3)_{r_L}^R \triangleq g_{p_R q_R r_L s_R}(a_{p_R}^\dagger a_{q_R}^\dagger a_{s_R}), \tag{A5}$$

$$(\hat{T}4)_{s_R}^L \triangleq g_{p_L q_L r_L s_R}(a_{p_L}^\dagger a_{q_L}^\dagger a_{r_L}), \tag{A6}$$

such that Eq. (A2) becomes

$$\begin{aligned}\hat{H}_2 = \hat{H}_2^L + \hat{H}_2^R &- g_{p_L q_R r_L s_R}(a_{p_L}^\dagger a_{r_L})(a_{q_R}^\dagger a_{s_R}) \\ &+ g_{p_L q_L r_R s_R}(a_{p_L}^\dagger a_{q_L}^\dagger)(a_{r_R} a_{s_R}) \\ &+ g_{p_R q_R r_L s_L}(a_{r_L} a_{s_L})(a_{p_R}^\dagger a_{q_R}^\dagger) + a_{p_L}^\dagger (\hat{T}1)_{p_L}^R \\ &+ (\hat{T}2)_{q_R}^L a_{q_R}^\dagger + a_{r_L}(\hat{T}3)_{r_L}^R + (\hat{T}4)_{s_R}^L a_{s_R}. \end{aligned} \tag{A7}$$

This reduces the bond dimensions for the last four terms in Eq. (A2) to $O(K)$. If we disregard the contributions from the one-body integrals, the last four terms involving the $\hat{T}1, \hat{T}2, \hat{T}3, \hat{T}4$ operators are equivalent to the terms involving the $\hat{S}_p^{L_k/R_k}$ operators in Eq. (30).

For the remaining two-body terms that involve two left and two right integral indices, the integrals can be collected with either the left or the right operators. However, regardless of the different choices, the bond dimension for the two-body Hamiltonian clearly scales as $O(K^2)$, although the actual distributions $D(k)$ across the partitions of the orbitals can be different as demonstrated below.

To analyze the different possibilities, we consider the recursion rules that link the right operators in Eq. (A7) to the next site. Let $R = CR'$, where C denotes the new site that is being added to the right block and R' denotes the remaining sites, then the complementary operators $\hat{T}1$ and $\hat{T}3$ defined on the right block R become

RIGHTSLINK

$$(\hat{T}1)^{CR'}_{p_L} = (\hat{T}1)^{C}_{p_L} + (\hat{T}1)^{R'}_{p_L} + (g_{p_L q_{R'} r_C s_C} a_{r_C} a_{s_C}) a^\dagger_{q_{R'}} + (g_{p_L q_C r_C s_{R'}} a^\dagger_{q_C} a_{r_C}) a_{s_{R'}} + g_{p_L q_C r_{R'} s_{R'}} a^\dagger_{q_C} (a_{r_{R'}} a_{s_{R'}}) + g_{p_L q_{R'} r_C s_{R'}} (-a_{r_C}) (a^\dagger_{q_{R'}} a_{s_{R'}}) \quad \text{(A8)}$$

and

$$(\hat{T}3)^{CR'}_{r_L} = (\hat{T}3)^{C}_{r_L} + (\hat{T}3)^{R'}_{r_L} + (-g_{p_C q_{R'} r_L s_C} a^\dagger_{p_C} a_{s_C}) a^\dagger_{q_{R'}} + (g_{p_C q_C r_L s_{R'}} a^\dagger_{p_C} a^\dagger_{q_C}) a_{s'_R} + g_{p_{R'} q_{R'} r_L s_C} a_{s_C} (a^\dagger_{p_{R'}} a^\dagger_{q_{R'}}) + g_{p_C q_{R'} r_L s_{R'}} a^\dagger_{p_C} (a^\dagger_{q_{R'}} a_{s_{R'}}), \quad \text{(A9)}$$

respectively. Similar to Eq. (A7), the integrals g_{pqrs} in the last lines of both Eqs. (A9) and (A8) can either be collected with the operators in C or R', without changing the leading bond dimension of $O(K^2)$ for $\hat{H}_2$. However, in order to maximally reuse common intermediates, the choice here for $\hat{T}1$ and $\hat{T}3$ also affects the assignment of integrals in Eq. (A7) for $\hat{H}_2$.

We first examine the case where the unassigned integrals in Eqs. (A7), (A8), and (A9) are all combined with the right operators. This is the choice of complementary operators as introduced in Eq. (35). In this case, the following complementary operators can be defined:

$$\hat{Q}^R_{p_L r_L} = g_{p_L q_R r_L s_R} (a^\dagger_{q_R} a_{s_R}), \quad \text{(A10)}$$

$$\underline{\hat{P}}^R_{p_L q_L} = g_{p_L q_L r_R s_R} (a_{r_R} a_{s_R}), \quad \text{(A11)}$$

$$\hat{\bar{P}}^R_{r_L s_L} = g_{p_R q_R r_L s_L} (a^\dagger_{p_R} a^\dagger_{q_R}), \quad \text{(A12)}$$

such that the related terms contributing to $\hat{H}_2$ (A7), $\hat{T}1$ (A8), and $\hat{T}3$ (A9) can be rewritten as

$$\hat{H}_2 \Leftarrow -(a^\dagger_{p_L} a_{r_L}) \hat{Q}^R_{p_L r_L} + (a^\dagger_{p_L} a^\dagger_{q_L}) \underline{\hat{P}}^R_{p_L q_L} + (a_{r_L} a_{s_L}) \hat{\bar{P}}^R_{r_L s_L}, \quad \text{(A13)}$$

$$(\hat{T}1)^{CR'}_{p_L} \Leftarrow a^\dagger_{q_C} \underline{\hat{P}}^{R'}_{p_L q_C} + (-a_{r_C}) \hat{Q}^{R'}_{p_L r_C}, \quad \text{(A14)}$$

$$(\hat{T}3)^{CR'}_{r_L} \Leftarrow a_{s_C} \hat{\bar{P}}^{R'}_{r_L s_C} + a^\dagger_{p_C} \hat{Q}^{R'}_{p_C r_L}, \quad \text{(A15)}$$

respectively. Meanwhile, since the expansions of the one-body terms $\hat{Q}$, $\underline{\hat{P}}$, and $\hat{\bar{P}}$ for $R = CR'$ do not require new intermediates, the recursion basis for the recursion to the rightmost site is complete and given by

$$\left(\hat{H}^R_2, a^{R\dagger}, a^R, (\hat{T}1)^R_L, (\hat{T}3)^R_L, \hat{Q}^R_{LL}, \underline{\hat{P}}^R_{LL}, \hat{\bar{P}}^R_{LL}, \hat{I}^R \right). \quad \text{(A16)}$$

(This basis corresponds to the elements of the vector in Eq. (37).) The leading bond dimension determined by the triple $(\hat{Q}^R_{LL}, \underline{\hat{P}}^R_{LL}, \hat{\bar{P}}^R_{LL})$ is $D_1(k) = O(k^2 + k^2/2 * 2) = O(2k^2)$ along the one-dimensional array of orbitals. The averaged value along all the sites is given by $\bar{D}_1 = 2/3K^2$.

Instead of using the complementary operators $(\hat{Q}^R_{LL}, \underline{\hat{P}}^R_{LL}, \hat{\bar{P}}^R_{LL})$, the integrals can also be collected with the left operators, viz.,

$$H_2 \Leftarrow (-g_{p_L q_R r_L s_R} a^\dagger_{p_L} a_{r_L})(a^\dagger_{q_R} a_{s_R}) + (g_{p_L q_L r_R s_R} a^\dagger_{p_L} a^\dagger_{q_L})(a_{r_R} a_{s_R}) + (g_{p_R q_R r_L s_L} a_{r_L} a_{s_L})(a^\dagger_{p_R} a^\dagger_{q_R}), \quad \text{(A17)}$$

$$(\hat{T}1)^{CR'}_{p_L} \Leftarrow (g_{p_L q_C r_{R'} s_R} a^\dagger_{q_C})(a_{r_{R'}} a_{s_{R'}}) + (-g_{p_L q_{R'} r_C s_{R'}} a_{r_C})(a^\dagger_{q_{R'}} a_{s_{R'}}), \quad \text{(A18)}$$

$$(\hat{T}3)^{CR'}_{r_L} \Leftarrow (g_{p_{R'} q_{R'} r_L s_C} a_{s_C})(a^\dagger_{p_{R'}} a^\dagger_{q_{R'}}) + (g_{p_C q_{R'} r_L s_{R'}} a^\dagger_{p_C})(a^\dagger_{q_{R'}} a_{s_{R'}}). \quad \text{(A19)}$$

With this choice, the basis for recursion to the rightmost site becomes

$$\left(\hat{H}^R_2, a^{R\dagger}, a^R, (\hat{T}1)^R_L, (\hat{T}3)^R_L, \hat{B}^R, \underline{\hat{A}}^R, \hat{\bar{A}}^R, \hat{I}^R \right), \quad \text{(A20)}$$

where the bare operators are defined as

$$\hat{B}^R_{q_R s_R} = a^\dagger_{q_R} a_{s_R}, \quad \text{(A21)}$$

$$\underline{\hat{A}}^R_{r_R s_R} = a_{r_R} a_{s_R}, \quad \text{(A22)}$$

$$\hat{\bar{A}}^R_{p_R q_R} = a^\dagger_{p_R} a^\dagger_{q_R}, \quad \text{(A23)}$$

which determines the leading bond dimension as $D_2(k) = O\left((K-k)^2 + (K-k)^2/2 * 2\right) = O\left(2(K-k)^2\right)$ and $\bar{D}_2 = 2/3K^2$.

Alternatively, mixed schemes that use different combinations of the pairs ($\hat{B}^R$-$\hat{Q}^R_{LL}$, $\underline{\hat{A}}^R$-$\underline{\hat{P}}^R_{LL}$, and $\hat{\bar{A}}^R$-$\hat{\bar{P}}^R_{LL}$) are also possible. For instance, the recursion using $\hat{B}^R$, $\underline{\hat{P}}^R_{LL}$, and $\hat{\bar{P}}^R_{LL}$ gives the basis for recursion as

$$\left(\hat{H}^R_2, a^{R\dagger}, a^R, (\hat{T}1)^R_L, (\hat{T}3)^R_L, \hat{B}^R, \underline{\hat{P}}^R_{LL}, \hat{\bar{P}}^R_{LL}, \hat{I}^R \right), \quad \text{(A24)}$$

which yields the leading bond dimension $D_3(k) = O\left((K-k)^2 + k^2/2 * 2\right) = O\left((K-k)^2 + k^2\right)$ and $\bar{D}_3 = 2/3K^2$. Similarly, the basis for the recursion using $\hat{Q}^R_{LL}$, $\underline{\hat{A}}^R$, and $\hat{\bar{A}}^R$ reads

$$\left(\hat{H}^R_2, a^{R\dagger}, a^R, (\hat{T}1)^R_L, (\hat{T}3)^R_L, \hat{Q}^R_{LL}, \underline{\hat{A}}^R, \hat{\bar{A}}^R, \hat{I}^R \right), \quad \text{(A25)}$$

and the leading bond dimension is also $D_3(k)$, the same as that for Eq. (A24).

The different distributions $D_{1,2,3}(k)$ discussed so far are compared in Figure 10 for $K = 50$. It is shown that the mixed schemes with $D_3(k)$ lead to a more balanced distribution of bond dimensions, although $D_1(k)$, $D_2(k)$, and $D_3(k)$ all share the same averaged value $2/3K^2$. Note also that the different recursions can also be changed at different sites, such as the central site $k = K/2$, resulting in a centrosymmetric distribution. The conventional DMRG algorithm using complementary operators employs this fact and uses the biased distributions $D_1(k)$ and $D_2(k)$. Specifically, DMRG follows $D_1(k)$ in the left part of sites and changes to $D_2(k)$ after the middle site, $k > K/2$. This gives the smallest computational cost in practice.

Next, we examine the MPO construction based on $\hat{H}_m = a^\dagger_m \hat{T}_m$ introduced in Eq. (51). The analysis for $\hat{T}_m$, with the first index fixed to be m, is very similar to that for

RIGHTSLINK

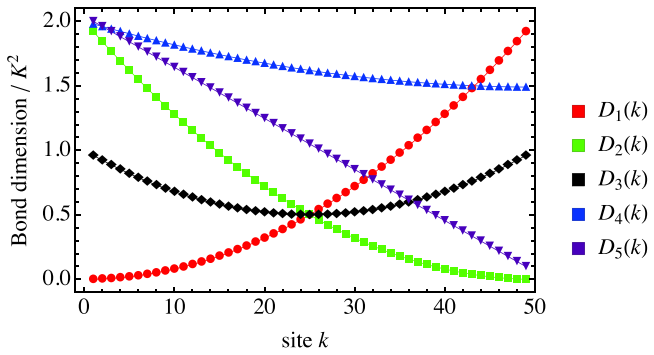

FIG. 10. Different distributions of the leading bond dimensions $D_n(k)$ ($n = 1, 2, 3, 4, 5$) for $K = 50$ derived based on the recursions in Eqs. (A16), (A20), (A24), (A26), and (A29), respectively.

$(\hat{T}1)^R_{p_L}$ in Eq. (A14), except that the index $p_L = m$ now runs through all sites. The recursion basis for $(\hat{T}1)^R_m$ can be deduced from Eq. (A14) as

$$\left((\hat{T}1)^R_m, a^{R\dagger}, a^R, \hat{Q}^R_{mL}, \underline{\hat{P}}^R_{mL}, \hat{I}^R\right), \tag{A26}$$

while Eq. (A18) for $(\hat{T}1)^R_m$ leads to a larger bond dimension with scaling $O(K^2)$. Thus, in such construction of MPO, there is no ambiguity for the choice of an optimal recursion basis. For given m, the leading bond dimension for Eq. (A26) can be found to be $D(m,k) = 2(K-k) + k + \begin{cases} k-m, & k > m \\ 0, & k \le m \end{cases}$, where the last m-dependent term arises from $\underline{\hat{P}}^R_{mL}$, which has nonzero contributions only for $m < k$. Thus, for each m the bond dimension for $(\hat{T}1)^R_m$ and hence $\hat{H}_m$ is of $O(K)$.

Although the separate sub-Hamiltonians $\hat{H}_m$ constitute separate MPO's which can be independently manipulated, to provide a point of comparison with the earlier distributions $D_{1,2,3}(k)$ for the single $\hat{H}$ MPO, we can compute the sum of the bond dimensions of all the sub-Hamiltonians, viz., $D_4(k) = \sum_{m=1}^K D(m,k) = O(2K^2 - kK + k^2/2)$. The averaged value of $D_4(k)$ is $\bar{D}_4 = 5/3K^2$ and the distribution for $K = 50$ is also shown in Figure 10. $D_4(k)$ is significantly larger then $D_{1,2,3}(k)$. This redundancy is due to the repeated use of $a^{R\dagger}$ and a^R in Eq. (A26) for all sub-Hamiltonians $\hat{H}_m$, while in the former case only a single instance of these operators is required in the recursion rules for $\hat{H}$ and thus they do not contribute multiple times to the leading bond dimension $D_{1,2,3}(k)$. Indeed, the increase of $\bar{D}_4 = 5/3K^2$ by K^2 as compared to $\bar{D}_{1,2,3} = 2/3K^2$ is attributable to these two terms, whose contribution to $\bar{D}_4$ is $1/K \sum_{k=1}(\sum_{m=1}^K 2(K-k)) = O(K^2)$. However, it is important to note that $D_4(k)$ does not constitute a true computational bond dimension, as in practice, the different $\hat{H}_m$ are manipulated separately and the combined bond dimension does not appears in an actual calculation.

As discussed in the main text, there is an alternative definition of the sub-Hamiltonian $\hat{H}_m$ as a collection of terms sharing the same rightmost operator. In this definition, the redundancy in the recursion rules from the repeated use of $a^{R\dagger}$ and a^R can be partially mitigated, as the "delocalization" of $a^{R\dagger}$ and a^R onto sites $k > m$ is removed. By this definition, the reduction of the (combined for all $\hat{H}_m$) bond dimension for each site index k can be estimated as follows: for $m < k$ the number of $a^{R\dagger}$ and a^R eliminated is $2(K-k)$, while for $m > k$ the number of necessary $a^{R\dagger}$ and a^R to represent $\hat{H}_m$ is only $2(m-k)$, and hence the number of unnecessary $a^{R\dagger}$ and a^R is $2(K-m)$. Then, the averaged reduction is $1/K \sum_{k=1}^K \left(\sum_{m=1}^k 2(K-k) + \sum_{m=k}^K 2(K-m)\right) = O(2/3K^2)$, and the averaged bond dimension for $\hat{H}_m$ becomes $5/3K^2 - 2/3K^2 = K^2$. We can explicitly demonstrate that the above estimates are correct. Since, by definition, $\hat{H}_m$ contains terms having at least one index on the site m, and hence, in terms of the bipartition, if the right block is the site m and the left block contains the sites from 1 to $m-1$, $\hat{H}_m$ collects all terms in Eq. (A2) except for $\hat{H}^L_2$. Thus, to analyze the bond dimension for $\hat{H}_m$, we can simply use the results for $\hat{H}_2$. Specifically, only the recursions for $(\hat{T}2)^L_{q_R}$ and $(\hat{T}4)^L_{s_R}$ from the site $m-1$ to the leftmost site is relevant to estimating the leading bond dimensions $D(k)$. Let $L = L'C$, the relevant recursions are found as

$$(\hat{T}2)^{L'C}_{q_R} \Leftarrow \underline{\hat{P}}^{L'}_{p_C q_R} a^\dagger_{p_C} + \hat{Q}^{L'}_{q_R s_C} a_{s_C}, \tag{A27}$$

$$(\hat{T}4)^{L'C}_{s_R} \Leftarrow \hat{Q}^{L'}_{q_C s_R}(-a^\dagger_{q_C}) + \hat{\bar{P}}^{L'}_{r_C s_R} a_{r_C}, \tag{A28}$$

where $\hat{Q}^L_{qs} \triangleq \sum_{pr \in L} g_{pqrs} a^\dagger_p a_r$. Thus, similar to Eq. (A16), the recursion basis is

$$\left((\hat{T}2)^L_m, (\hat{T}4)^L_m, a^{L\dagger}, a^L, \hat{Q}^L_{mR}, \hat{Q}^L_{Rm}, \underline{\hat{P}}^L_{Rm}, \hat{\bar{P}}^L_{Rm}, \hat{I}^R\right), \tag{A29}$$

where R denotes the sites between k and m in this expression. The leading bond dimension becomes $D(m,k) = 2k + 2(m-k) + 2(m-k) = 2(2m-k)$ for $k < m$, while the bond dimension for $k > m$ is simply 1. Thus it is seen that $\hat{H}_m$ defined in this way also has a bond dimension of $O(K)$. The leading bond dimension $D_5(k)$ obtained by summing over m can be estimated as $D_5(k) = \sum_{m=1}^K D(m,k) = \sum_{m=k}^K D(m,k) = O(2K^2 - 2kK)$, which decays linearly with the increase of k. Its averaged value is $\bar{D}_5 = K^2$, which agrees with our estimates from the consideration of redundancies. This value is much smaller than $\bar{D}_4 = 5/3K^2$, but larger than $\bar{D}_{1,2,3} = 2/3K^2$. The distribution $D_5(k)$ is displayed in Figure 10. While $D_5(k)$ represents a combined bond dimension and does not directly reflect the computational structure where the $\hat{H}_m$ are manipulated separately, we nonetheless expect there will be a computational gain from using this second definition of $\hat{H}_m$ compared to the use of $\hat{H}_m$ in Eq. (51) in practical computations.

[1]S. R. White, Phys. Rev. Lett. **69**, 2863 (1992).
[2]S. R. White, Phys. Rev. B **48**, 10345 (1993).
[3]T. Xiang, Phys. Rev. B **53**, R10445 (1996).
[4]S. R. White and R. L. Martin, J. Chem. Phys. **110**, 4127 (1999).
[5]S. Daul, I. Ciofini, C. Daul, and S. R. White, Int. J. Quantum Chem. **79**, 331 (2000).
[6]G. K.-L. Chan and M. Head-Gordon, J. Chem. Phys. **116**, 4462 (2002).
[7]G. K.-L. Chan, J. Chem. Phys. **120**, 3172 (2004).
[8]Ö. Legeza, J. Röder, and B. Hess, Phys. Rev. B **67**, 125114 (2003).
[9]Ö. Legeza, J. Röder, and B. Hess, Mol. Phys. **101**, 2019 (2003).
[10]A. O. Mitrushenkov, G. Fano, F. Ortolani, R. Linguerri, and P. Palmieri, J. Chem. Phys. **115**, 6815 (2001).
[11]G. K.-L. Chan and M. Head-Gordon, J. Chem. Phys. **118**, 8551 (2003).
[12]Ö. Legeza and J. Sólyom, Phys. Rev. B **68**, 195116 (2003).
[13]Ö. Legeza and J. Sólyom, Phys. Rev. B **70**, 205118 (2004).
[14]A. Mitrushenkov, R. Linguerri, P. Palmieri, and G. Fano, J. Chem. Phys. **119**, 4148 (2003).
[15]G. K.-L. Chan, M. Kállay, and J. Gauss, J. Chem. Phys. **121**, 6110 (2004).

RIGHTSLINK

[16]G. K.-L. Chan and T. Van Voorhis, J. Chem. Phys. **122**, 204101 (2005).
[17]G. Moritz and M. Reiher, J. Chem. Phys. **124**, 034103 (2006).
[18]J. Hachmann, W. Cardoen, and G. K.-L. Chan, J. Chem. Phys. **125**, 144101 (2006).
[19]K. H. Marti, I. M. Ondík, G. Moritz, and M. Reiher, J. Chem. Phys. **128**, 014104 (2008).
[20]D. Ghosh, J. Hachmann, T. Yanai, and G. K.-L. Chan, J. Chem. Phys. **128**, 144117 (2008).
[21]G. K.-L. Chan, Phys. Chem. Chem. Phys. **10**, 3454 (2008).
[22]D. Zgid and M. Nooijen, J. Chem. Phys. **128**, 144115 (2008).
[23]K. H. Marti and M. Reiher, Z. Phys. Chem. **224**, 583 (2010).
[24]H.-G. Luo, M.-P. Qin, and T. Xiang, Phys. Rev. B **81**, 235129 (2010).
[25]K. H. Marti and M. Reiher, Phys. Chem. Chem. Phys. **13**, 6750 (2011).
[26]Y. Kurashige and T. Yanai, J. Chem. Phys. **135**, 094104 (2011).
[27]S. Sharma and G. K.-L. Chan, J. Chem. Phys. **136**, 124121 (2012).
[28]G. K. Chan, Wiley Interdiscip. Rev.: Comput. Mol. Sci. **2**, 907 (2012).
[29]S. Wouters, P. A. Limacher, D. Van Neck, and P. W. Ayers, J. Chem. Phys. **136**, 134110 (2012).
[30]W. Mizukami, Y. Kurashige, and T. Yanai, J. Chem. Theory Comput. **9**, 401 (2012).
[31]Y. Kurashige, G. K.-L. Chan, and T. Yanai, Nat. Chem. **5**, 660 (2013).
[32]S. Sharma, K. Sivalingam, F. Neese, and G. K.-L. Chan, Nat. Chem. **6**, 927 (2014).
[33]S. Wouters and D. Van Neck, Eur. Phys. J. D **68**, 1 (2014).
[34]S. Wouters, W. Poelmans, P. W. Ayers, and D. Van Neck, Comput. Phys. Commun. **185**, 1501 (2014).
[35]E. Fertitta, B. Paulus, G. Barcza, and Ö. Legeza, Phys. Rev. B **90**, 245129 (2014).
[36]S. Knecht, Ö. Legeza, and M. Reiher, J. Chem. Phys. **140**, 041101 (2014).
[37]S. Szalay, M. Pfeffer, V. Murg, G. Barcza, F. Verstraete, R. Schneider, and Ö. Legeza, Int. J. Quantum Chem. **115**, 1342 (2015).
[38]T. Yanai, Y. Kurashige, W. Mizukami, J. Chalupskỳ, T. N. Lan, and M. Saitow, Int. J. Quantum Chem. **115**, 283 (2015).
[39]R. Olivares-Amaya, W. Hu, N. Nakatani, S. Sharma, J. Yang, and G. K.-L. Chan, J. Chem. Phys. **142**, 034102 (2015).
[40]Z. Li and G. K.-L. Chan, J. Chem. Phys. **144**, 084103 (2016).
[41]S. Guo, M. A. Watson, W. Hu, Q. Sun, and G. K.-L. Chan, J. Chem. Theory Comput. **12**, 1583 (2016).
[42]S. Östlund and S. Rommer, Phys. Rev. Lett. **75**, 3537 (1995).
[43]S. Rommer and S. Östlund, Phys. Rev. B **55**, 2164 (1997).
[44]F. Verstraete, J. J. Garcia-Ripoll, and J. I. Cirac, Phys. Rev. Lett. **93**, 207204 (2004).
[45]I. P. McCulloch, J. Stat. Mech.: Theory Exp. **2007**, P10014.
[46]F. Verstraete, V. Murg, and J. I. Cirac, Adv. Phys. **57**, 143 (2008).
[47]B. Pirvu, V. Murg, J. I. Cirac, and F. Verstraete, New J. Phys. **12**, 025012 (2010).
[48]G. Vidal, Phys. Rev. Lett. **93**, 040502 (2004).
[49]A. J. Daley, C. Kollath, U. Schollwöck, and G. Vidal, J. Stat. Mech.: Theory Exp. **2004**, P04005.
[50]S. R. White and A. E. Feiguin, Phys. Rev. Lett. **93**, 076401 (2004).
[51]G. Vidal, Phys. Rev. Lett. **98**, 070201 (2007).
[52]R. Orus and G. Vidal, Phys. Rev. B **78**, 155117 (2008).
[53]I. P. McCulloch, preprint arXiv:0804.2509 (2008).
[54]A. E. Feiguin and S. R. White, Phys. Rev. B **72**, 220401 (2005).
[55]T. Nishino, Y. Hieida, K. Okunishi, N. Maeshima, Y. Akutsu, and A. Gendiar, Prog. Theor. Phys. **105**, 409 (2001).
[56]F. Verstraete and J. I. Cirac, preprint arXiv:cond-mat/0407066 (2004).
[57]R. Orús, Ann. Phys. **349**, 117 (2014).
[58]V. Murg, F. Verstraete, Ö. Legeza, and R. Noack, Phys. Rev. B **82**, 205105 (2010).
[59]N. Nakatani and G. K.-L. Chan, J. Chem. Phys. **138**, 134113 (2013).
[60]V. Murg, F. Verstraete, R. Schneider, P. Nagy, and O. Legeza, J. Chem. Theory Comput. **11**, 1027 (2015).
[61]U. Schollwöck, Ann. Phys. **326**, 96 (2011).
[62]N. Nakatani, "MPSXX: Mps/mpo multi-linear library implemented in c++11," https://github.com/naokin/mpsxx.
[63]S. Keller and M. Reiher, CHIMIA Int. J. Chem. **68**, 200 (2014).
[64]S. Keller, M. Dolfi, M. Troyer, and M. Reiher, J. Chem. Phys. **143**, 244118 (2015).
[65]S. Keller and M. Reiher, J. Chem. Phys. **144**, 134101 (2016).
[66]E. Stoudenmire and S. R. White, ITensor: A c++ library for creating efficient and flexible physics simulations based on tensor product wavefunctions, http://itensor.org/.
[67]M. Dolfi, B. Bauer, S. Keller, A. Kosenkov, T. Ewart, A. Kantian, T. Giamarchi, and M. Troyer, Comput. Phys. Commun. **185**, 3430 (2014).
[68]S. Sharma and G. K.-L. Chan, J. Chem. Phys. **141**, 111101 (2014).
[69]S. Sharma and A. Alavi, J. Chem. Phys. **143**, 102815 (2015).
[70]S. Sharma, G. Jeanmairet, and A. Alavi, J. Chem. Phys. **144**, 034103 (2016).
[71]J. J. Dorando, J. Hachmann, and G. K.-L. Chan, J. Chem. Phys. **130**, 184111 (2009).
[72]N. Nakatani, S. Wouters, D. Van Neck, and G. K.-L. Chan, J. Chem. Phys. **140**, 024108 (2014).
[73]G. K.-L. Chan, J. J. Dorando, D. Ghosh, J. Hachmann, E. Neuscamman, H. Wang, and T. Yanai, *Frontiers in Quantum Systems in Chemistry and Physics* (Springer Netherlands, 2008), pp. 49–65.
[74]G. K.-L. Chan and S. Sharma, Annu. Rev. Phys. Chem. **62**, 465 (2011).
[75]G. Hager, E. Jeckelmann, H. Fehske, and G. Wellein, J. Comput. Phys. **194**, 795 (2004).
[76]Y. Kurashige and T. Yanai, J. Chem. Phys. **130**, 234114 (2009).
[77]J. Rincón, D. García, and K. Hallberg, Comput. Phys. Commun. **181**, 1346 (2010).
[78]E. Stoudenmire and S. R. White, Phys. Rev. B **87**, 155137 (2013).
[79]C. Nemes, G. Barcza, Z. Nagy, Ö. Legeza, and P. Szolgay, Comput. Phys. Commun. **185**, 1570 (2014).

RIGHTSLINK